Data Mining of Many-Attribute Data: Investigating the Interaction between Feature Selection Strategy and Statistical Features of Datasets

By

Silang Luo

Submitted for the degree of Doctor of Philosophy

Heriot Watt University, Edinburgh

Computer Science – MACS

<06> <2009>

ABSTRACT

In many datasets, there is a very large number of attributes (e.g. many thousands). Such datasets can cause many problems for machine learning methods. Various feature selection (FS) strategies have been developed to address these problems. The idea of an FS strategy is to reduce the number of features in a dataset (e.g. from many thousands to a few hundred) so that machine learning and/or statistical analysis can be done much more quickly and effectively. Obviously, FS strategies attempt to select the features that are most important, considering the machine learning task to be done. The work presented in this dissertation concerns the comparison between several popular feature selection strategies, and, in particular, investigation of the interaction between feature selection strategy and simple statistical features of the dataset. The basic hypothesis, not investigated before, is that the correct choice of FS strategy for a particular dataset should be based on a simple (at least) statistical analysis of the dataset.

First, we examined the performance of several strategies on a selection of datasets. Strategies examined were: four widely-used FS strategies (*Correlation*, *Relief F*, *Evolutionary Algorithm*, *no-feature-selection*), several *feature bias* (FB) strategies (in which the machine learning method considers all features, but makes use of bias values suggested by the FB strategy), and also combinations of FS and FB strategies. The results showed us that FB methods displayed strong capability on some datasets and that combined strategies were also often successful.

Examining these results, we noted that patterns of performance were not immediately understandable. This led to the above hypothesis (one of the main contributions of the thesis) that statistical features of the dataset are an important consideration when choosing an FS strategy. We then investigated this hypothesis with several further experiments. Analysis of the results revealed that a simple statistical feature of a dataset, that can be easily pre-calculated, has a clear relationship with the performance of certain FS methods, and a similar relationship with differences in performance between certain pairs of FS strategies.

In particular, Correlation based FS is a very widely-used FS technique based on the basic hypothesis that good feature sets contain features that are highly correlated with the class, yet uncorrelated with each other. By analysing the outcome of several FS strategies on different artificial datasets, the experiments suggest that CFS is never the best choice for poorly correlated data.

Finally, considering several methods, we suggest tentative guidelines for choosing an FS strategy based on simply calculated measures of the dataset.

ACKNOWLEDGEMENTS

I would like express my gratitude and thanks to

- Professor David W Corne. My Only supervisor, for encouraging me to undertake the research, for providing continuous support. I would also like thank him for his unconditional trust and motivation.

- My parents (Xiaorong Lü and Xiaochuan Luo) for their generous support and necessary funds. I would also like to express my gratitude for their valuable love and willingness.

- MACS School of Heriot Watt University, for good research environment.

Thank you all.

Contents

LISTS OF FIGURES

LISTS OF TABLES

LISTS OF PUBLICATIONS

Luo S and Corne D W (2008). Feature Selection Strategies for Poorly Correlated Data: Correlation Coefficient Considered Harmful, in Kazovsky *et al.* (eds.), *Proceedings of the 7th WSEAS Int'l Conf. on Artificial Intelligence, Knowledge Engineering and Databases* (AIKED'08), Cambridge, UK, WSEAS Press, pp. 226—231, ISBN: 978-960-6766-41-1, ISSN: 1790-5109.

Luo S and Corne D W (2009). Comparing Feature Bias and Feature Selection Strategy for Many-Attribute machine learning, (Submitted for publication in 4th International Symposium on Intelligence Computation and Applications (ISICA 2009)).

Luo S and Corne D W (2009). Data Mining of Many-Attribute Data: Investigating the Interaction between Feature Selection (bias) Strategies and Statistical Features of Datasets. (To be submitted to international Journal of Computational Intelligence in Bioinformatics and Systems Biology, 2009).

Chapter 1

Introduction and Background

1.1 Introduction

During recent decades, a prominent characteristic of molecular biology has been the rapidly expanding amount of biological data. Thus, a growing problem is presented to scientists: how to accurately interpret and make full use of the growing amount of information. For example, if we could understand the structure and the function of expressed proteins, it may be a breakthrough in diagnosis and treatment for a specific disease.

Machine learning, in general, is a tool that can analyse large quantities of data automatically and play a significant role in such breakthroughs. Within machine learning, and especially when we are concerned with certain types of biological data, Feature Selection (FS) (Guyon I and Elisseeff A (2003)) is a key aspect. This involves minimizing the number of features that we consider in a dataset, but still attempting to maximise the predictive power of the model that we build by doing machine learning on the dataset. Feature selection is a common task in many classification and regression problems; it is necessary because machine learning tools often cannot cope when the data has thousands of attributes. This is quite common in bioinformatics data, and we therefore focus this work on bioinformatics data.

The hypothesis explored in this thesis is that: an appropriate choice of FS method for a dataset can be determined by first calculating simple statistical measures of that dataset. Broadly speaking, we test this hypothesis by experimenting with several different FS methods, and then running a machine learning method on the dataset using only the selected features. The result of the machine learning method is an accuracy value (for example, the accuracy of predicting the target class on a test set). This accuracy value can be considered as an evaluation of the suitability of the FS method that was used. We then examine the relationship between the accuracy of the FS method and a simple statistical feature of the dataset. Mainly we use the *highest statistical correlation between a feature and the target class* as the simple statistical characteristic of the dataset, and we call this the dataset correlation value (DCV). As we will see, by investigation of simple statistical correlation based feature selection (CFS) (Hall M A (2000)) and other popular methods on specific data, when the DCV is quite low, the experiments indicate that CFS is never the best FS method, and it is sometimes the worst. This is intuitively reasonable, but has not been highlighted in the research literature, and there are many examples of cases where researches use CFS as the FS method despite the dataset having a low DCV.

In the remainder of this introductory chapter, Section 1.2 broadly introduces the topics of machine learning and classification. Then we will discuss the issues and problems of large-scale data in section 1.3. Section 1.4 presents a brief introduction to bioinformatics and related data examined in this thesis. In section 1.5 we provide an overview of the contributions of this thesis, and then in Section 1.6 we provide a general overview of the contents of the remaining chapters of the thesis.

1.2 Machine Learning

A program learns from experience E with respect to some class of tasks T and performance measure P, if its performance at task T, as measured by P, improves with experience E.

---- Mitchell T (1997)

Machine learning research is motivated partly by how to create machines to simulate the act of human learning, so that machines could get new knowledge or skills, and then reorganise the structure of their previous knowledge to improve performance. For the moment, the most common applied definition of machine learning is the study of computational methods for improving automatically the performance of machines through experience.

1.2.1 What is Learning?

Learning is like intelligence, covering such a broad range of processes that it is difficult to define precisely. Michalski R S and Kodratoff Y (1990) view learning as a process of modifying the learner’s knowledge by exploring the learner’s experience. Zoologists and psychologists study learning in animals and humans and define learning as “to gain knowledge, or understanding of, or skill in by study instruction or experience” (Nilsson N J (1996)). As the core of any learning agent, whether the agent is animal or mechanical or software, is an algorithm that defines the process that is used for learning. When an agent learns, it acquires knowledge. Learning is a process that allows an agent to adapt its performance through instruction or experience.

Learning occurs when a system makes accurate generalisations about the problem domain. Information must be assimilated from the environment, and then the internal representation of the domain must be modified to accommodate the new data relative to what is already known. Over a sufficiently large set of training examples, the system should be able to generalise the problem space for unknown examples (induction; reasoning from the particular to the general).

The motivation of the learning algorithm is to transform input data into a particular form of useful output. The output could be, for example, recognition of optically scanned handwritten text, or the next moves in a game of chess. This outcome of learning is often called the target function. If learning is successful, the target function should be able to take input data and produce a correct output.

1.2.2 History of Machine Learning

Machine learning is a subdivision of artificial intelligence (Rich E and Knight K (1991)). Its application widely extends over all the artificial intelligence areas. It usually uses induction and integration rather than deduction. A machine or software tool would not be viewed as intelligent unless it could adapt to change with the environment. However, early artificial intelligence systems showed a lack of learning ability. From the 1920s to the 1970s, artificial intelligence research was mostly based on deduction. People thought once machines would have the ability of logical deduction, the machine would have "intelligence". The outstanding work in this period was the logic theory machine by Newell A and Simon H A (1956). Because of that work, they won the Turing award in 1975. In the mid 1970s, a lot of expert systems contributed in this area. However, people recognised it was a difficult process to teach a machine knowledge which is summarised by a human. As a result, some experts considered that machines could learn knowledge by themselves. In the 1980s, most research and most applied work in this area were based on induction (learning from examples) (Elio R and Watanabe L (1991)).

Michalski R S, *et al.* (1983) divides the machine learning areas into six typical classes: 1) rote learning (Li X (2007)); 2) Learning from instruction; 3) Learning by deduction; 4) Learning by analogy; 5) Explanation-based learning; 6) Learning from induction.

An important distinction within these classes is the difference between supervised learning and unsupervised learning. In the next section, we briefly review these two kinds of learning methods.

1.2.3 Supervised Learning and Unsupervised Learning

In supervised machine learning (Dougherty D, Kohavi R and Sahami M (1995)), there is always the concept of target labels. For example, some data instances may be labelled "cancer", and other data instances labelled "not cancer". In supervised machine learning, the algorithms take externally supplied instances and their labels (usually called training instances or the training set), and try to produce general hypotheses, which then make predictions about the labels of future instances. In other words, the goal of supervised learning is to build a good model of the distribution of class labels in terms of features. Every instance in any dataset used by machine learning algorithms is represented using the same set of features. The features may be continuous, categorical or binary. When instances are given with known labels (the corresponding correct outputs) then the learning is called supervised.

In unsupervised learning systems, in contrast to supervised learning, the instances are without labels. Often the goal in unsupervised learning is to decide which objects should be grouped together—in other words, the learner forms the classes itself. In the absence of any specific guidance, these systems attempt to discover patterns in the data. For example, clustering is a very common method for unsupervised learning.

Another kind of machine learning is reinforcement learning (Barto A G and Sutton R S (1997)). The training information provided to the learning system by the environment (which is considered to be an external trainer) is in the form of a scalar reinforcement signal that is a measure of how well the system operates. This results in a reward and the agent attempts to learn a policy, a general way to operate, for maximising this reward.

1.2.4 Classification Techniques

Classification is the process of assigning samples to a set of defined class labels. There has been significantly more research carried out into binary classification (members and non-members of a class) than multi-class classification.

Classification is a very common task in biological problems where given two different sets of examples, namely positive and negative examples, the learner needs to construct a classifier to distinguish between the positive examples and the negative set. This classifier can then be used as the basis for classifying as yet unseen data in the future. Usually, for a supervised classification problem, the training examples are in the form of a set of tuples where x_i is the class label and y_i is the set of attributes for the instances. The task of the learning algorithm is to produce a classifier (hypothesis, function) to classify the instances into the correct class.

Classification problems are represented by sets of samples from the problem domain, known as cases or instances, which each consist of a set of features or attributes. For example, a set of features might be a number of measurements of a botanical sample, and the class values might be species. Usually, when we do machine learning to try to learn a model that predicts the class value, most or all of the features in the dataset are *relevant* to the task. That is, we might expect that each feature is needed in order to produce an effective model. But, in the area of bioinformatics especially, there are many datasets now where the following two things are true: (1) the dataset contains many thousands of features; (2) we have little or no *a priori* understanding about which features are relevant, and in fact we could expect only a small percentage of them to be relevant.

This leads to a need for feature selection (FS) methods, and leads us to think about subsets of features. An optimal subset of features would be the smallest subset that can be found that leads to an accurate predictive model. While, for most classification problems, optimal subsets will contain relevant features, some care has to be taken when linking relevance automatically to optimal feature subsets. In the research of Kohavi R and John G H (1997), some examples are given, illustrating the fact that relevance of a feature does not necessarily imply that it is in the optimal feature subset.

1.3 Problems with Large-scale Datasets

As the amount of data stored in databases continues to grow fast, the analysis of large-scale datasets (Almuallim H and Dietterich T G (1991)) is crucial to uncovering important relationships. The datasets provide a wealth of information on the relevant system. The valuable hidden knowledge in these datasets could be used to improve many decision-making and similar processes. For example, in business and commerce**,** interesting relationships between customers and products might be contained in the database of previous sales; the key elements of mass spectrometry[1] might exhibit the relationship between the patients and survivors. Unfortunately, the ability to understand and make use of this information does not keep in pace with its growth. These problems, which are focussed on large-scale datasets, have become increasingly important.

What kind of knowledge we should try to discover in large-scale datasets? The basic idea is to build predictive models, so that we can predict the value of some important attribute (e.g. monthly sales, presence of cancer, etc…) based on the values of other attributes, In this context, our major task is to discover knowledge to help produce a high predictive accuracy rate. Moreover, if our predictive model is comprehensible for the user, we think this knowledge could be compatible and add to human knowledge. In many applications, comprehensible models are necessary. It is important that decisions made on the basis of the model can be understood and rationalised by humans. A popular way to have models that are also comprehensible, and this is the method we use in our experiments, is to use a set of IF-THEN (prediction) rules, where each rule is of the form:

```
IF <conditions are satisfied>
            THEN <predict value for attribute>
```

In the knowledge discovery process, the machine learning algorithm and the pre-processing part can be considered as vital steps as seen in Figure 1.1.

The question of how to choose which algorithm is most suitable for a given dataset is getting more important for scientists.

The first step, Data integration involves, collecting and organizing the data from several different sources.

[1] Mass spectrometry, which is being applied to the measurement of DNA, RNA, protein and small molecule metabolites, is a key technology for the measurement of molecular structure and molecular abundance.

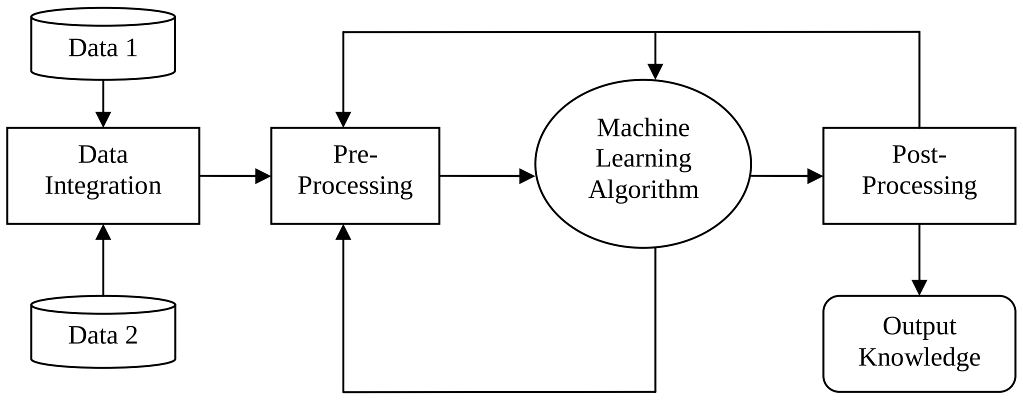


Figure 1.1. Overview of the knowledge discovery process.

The second step, Pre-processing, can consist of many activities. Common examples include reducing the noise in data (Quinlan J R (1989)), choosing a strategy for handling missing (unknown) feature values(Bruha I and Franel F (1996)), and selecting and ordering features according to their information contents. Our research focuses on how to select a subset of attributes relevant for classification. This process is motivated by the fact that the rules discovered by a machine learning algorithm should only contain just a few attributes; it is well known that this tends to lead to better accuracy. Hence, we first select an attribute subset and then provide only those selected attributes for the machine learning algorithm. Considering an example, suppose we try to predict the relationship between customer and products, and suppose there is an attribute named "Customer Name". A specific rule could be (IF Customer Name = <<a specific name>> then product = <<a specific product>>). This type of rule usually has no predictive power. Technically speaking, it is over-fitting[2] the data.

The third step, called Data mining, is to apply the machine learning algorithm to the reduced dataset or the dataset with selected attributes. Typically this results in a collection of predictive rules.

[2] Over-fitting: A classifier that performs well on the training examples but poorly on unseen instances.

The last step, Post-processing, is to extract a subset of interesting rules, among all discovered ones and simplify the large rule set, in order to improve knowledge comprehensibility for the user. "Entities must not be multiplied beyond necessity” by Occam's razor.

The knowledge discovery process is inherently iterative, and the output of a set can be sent as feedback to a previous step. This partly illustrates the distinction between filter and wrapper methods (which will be widely discussed in the next chapter). If the output of a step is used to influence a previous step in a next iteration, then we are using a wrapper-based approach; otherwise, it is a filter approach.

The difficult question here is how to avoid the process being affected by irrelevant attributes. This could be thought as how to choose a good feature selection (FS) algorithm for the data. Various FS and machine learning strategies have been developed; but this serves partly to make the problem worse: given a new large-scale dataset with perhaps thousands of features, it is a tough decision to choose the best FS strategies, especially since they usually come with little or no guidance.

Are combined strategies better than a single approach? A lot of experiments described in later chapters showed us that combined methods often perform better. Dietterich T G (2000) suggests three main reasons: statistical, computational and representational, to explain why no individual methods can claim that it is superior to others. As an example, Wu Y and Zhang A (2004) present an efficient feature selection method to facilitate classifying high-dimensional numerical data. Their method employs balanced information gain to measure the contribution of each feature (for data classification); and it calculates feature correlation with a novel extension of balanced information gain. Then they use a forward sequential selection algorithm to select uncorrelated features with large balanced information gain. This filter approach significantly improves the accuracy and efficiency. In our research, we implemented several combined strategies and compare with the original individual strategies on the same datasets. The experiments and analysis will be described in chapters 4 and 5.

Most large-scale datasets that have appeared recently come from the science of bioinformatics. Biological data are being produced at a phenomenal rate (Reichhardt T, *et al.* (1999)). For example as of August 2000, the GenBank repository of nucleic acid sequences contained 8,214,000 entries (Benson D A, *et al.* (2000)) and the SWISS-PROT database of protein sequences contained 88,166 (Bairoch A, *et al.* (2000)). On average, these databases are doubling in size every 15 months.

There are many other areas, apart from bioinformatics, where feature selection is an important concern. Just one example is text categorization, which is a domain where the datasets usually have a very large number of features. The features of examples to be classified are words, and the number of different words can be hundreds of thousands. However, an initial pruning of the most and least frequent words may reduce the effective number of words. While some simple document classification tasks can be accurately performed with vocabulary sizes of less than one hundred, many complex tasks on real-world data from the Web, UseNet and newswire articles do best with vocabulary sizes in the thousands (McCallum A and Nigam K (1998)). Typical tasks include the automatic sorting of URLs into a web directory and the detection of unsolicited email.

In the next section, we will introduce the bioinformatics and some related areas.

1.4 Bioinformatics

1.4.1 Introduction

Bioinformatics is the science of understanding and organising biological information by applying computational techniques. The application of computational techniques encompasses a wide range of subject areas including structural biology, genomics, proteomics, metabolomics, and more.

Bioinformatics (Huerta M, *et al*. (2000)), emphasising algorithm design and theoretical methods over the application of the technology, is not just the combination of computer science and biology. There are many areas of research (e.g. developing mathematical algorithms or statistical methodologies) that are relevant for analyzing data and thus uncovering biological knowledge. Bioinformatics includes the latter techniques, and it also has the role to provide the necessary computational and statistical means for data handling capabilities (Yu U, Lee S H, Kim Y J and Kim S (2004)). It is certainly not as simple as "number-crunching for molecular biologists", but is about the application of many techniques such as modelling, simulation, data abstraction, data manipulation and pattern discovery techniques.

According to Luscombe N M (2001), bioinformatics has three aims. First, it allows researchers to access existing information and to submit new information as it is produced. The second aim is to develop tools and resources that aid in the analysis of data. The third aim is to use these tools to analyse the data and interpret the results in a biologically meaningful manner.

Bioinformatics is a relatively young field, and the pace of research is driven by the large and rapidly increasing amount of data being produced from, for example, efforts to sequence the genomes of a variety of organisms. The data generated by the experimental scientists requires annotation and detailed analysis in order to turn it into knowledge which can then be applied to improving health care via, for example, new drugs and gene therapy, medical practices, and food production - all of which are now high-profile issues nationally.

1.4.2 Systems Biology

Systems biology studies biological systems by systematically perturbing them (biologically, genetically, or chemically); monitoring the gene, protein, and informational pathway responses; integrating these data; and as an end result, formulating mathematical models that try to describe the structure of the biological system and its response to individual perturbations (Ideker T *et al.* (2000) and Ideker

T *et al.* (2001)). Systems biology is the process involved in understanding the interactions and relationships between many parts of biological systems. It aims to look at biological systems as wholes, rather than specific or certain parts of a cell or organism. Top-down systems biology identifies molecular interaction networks on the basis of correlated molecular behaviour observed in genome-wide "omics" studies. Bottom-up systems biology examines the mechanisms through which functional properties arise in the interactions of known components (Bruggeman F J and Westerhoff H V (2007)).

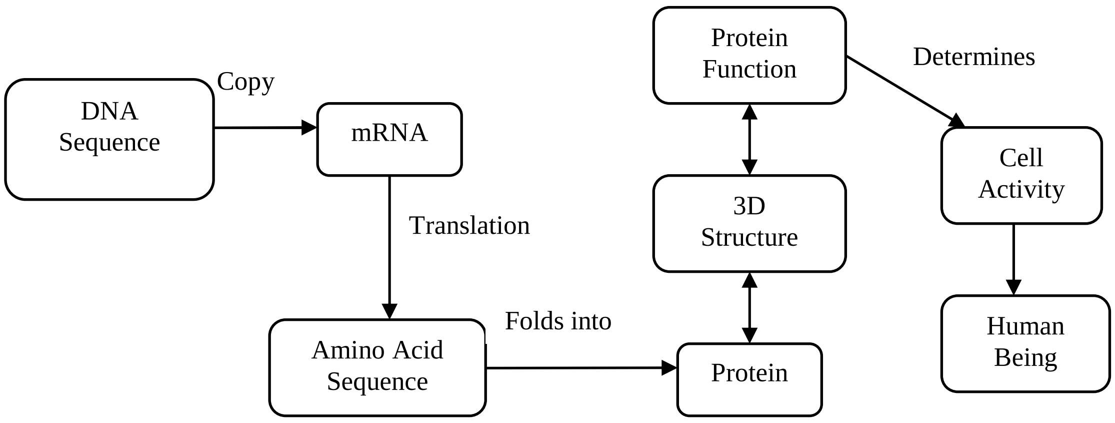


Figure 1.2. The process of system biology.

As hinted at by Figure 1.2, all biological information is hierarchical. Initially DNA will change over to mRNA, which in turn is translated into amino acids within ribosomes. Proteins are built by blocks of amino acids, and these then lead to protein interactions and proteins performing particular functions; in turn this leads to some informational pathways. These pathways form informational networks, which in turn become cells. Now cells form networks of cells. Finally an individual is a collection of cells. A host of individuals forms a population, and a variety of populations becomes an ecology. The primary challenge for biology and medicine is to create tools and mechanisms to capture and integrate these different levels of biological information towards gaining insight into their curious functions.

However, most of the biological data so far gathered are qualitative rather than quantitative and probably many breakthroughs in experimental devices, advanced software, and analytical methods are required before the achievements of systems biology can live up to their potential (Kitano H (2002)). Nevertheless, systems biology is needed and modelling approaches are powerful. Model building, as an aid to understand complex systems, is also the most preferred methodology in other areas, like ecology or economics. The overall promise is worth the effort, and lessons from system analysis of advanced technologies and engineering theory suggest that the system can be usefully divided into subsystems, so one does not have to tackle and solve the whole system at once.

1.4.3 Overview of Proteomics

Proteomics **(**Pandey A and Mann M (2000)) is defined as the large-scale study of the complete complement of proteins within a cell or tissue sample. It is an attempt to describe or explain their structures, function and expression of protein content under different conditions (stressed, diseased, and/or drugged) to further understand different biological processes. One of the main goals of proteomics is the identification of novel markers that can be used for prediction, prevention, diagnosis, prognosis and therapy optimization in human diseases. Similar to other "-omics", the development of proteomics was significantly influenced by the recent developments in technology. The real development of proteomics started only after the use of the two-dimensional (2D) protein electrophoresis method (Celis J E, Bravo R (Eds) (1984)) followed by mass spectrometry (MS) (Aebersold R and Mann M (2003)).

The 2-D gel electrophoresis (2D-GE) method, which enables to distinguish up to 10,000 proteins from a cell sample, is now the preferred method for protein separation. It is based on two distinct physical and chemical features of proteins: first, according to the isoelectric point (which is indicated by a pH value), the proteins in a sample are separated. Proteins then migrate towards the anode according to their total charge up to the point where the gel pH equals the pI of a given protein. Then, common electrophoresis on a polyacrylamide gel (PAGE) is applied, but the electric current is applied at a perpendicular angle to the original orientation of the electrodes. Proteins then migrate in the second direction through the gel only according to their size. In short, 2D-GE is used to first separate the proteins by isoelectric point and then by size.

Mass Spectrometry is an essential technique in the analysis of proteins and other biological molecules by virtue of their versatility, sensitivity, speed, and improving ease of use. It enables a scientist to localise modifications within a protein and also helps to find out the nature of such modifications. A mass spectrometer can be divided into three fundamental parts, namely the *ionisation source*, the *analyser*, and the *detector***.** The sample firstly is introduced into the ionisation source and the molecules in the sample are ionised, then these ions are extracted into the analyser region according to their mass-to-change ratios (m/z). Ultimately, a data system collects the signals from separated ions.

Through these 2D-GE and MS techniques, much research will be done increasingly in the future, to try to identify differentially expressed proteins for early diagnosis and treatment of specific diseases. Differential expression, for example, means that machine learning methods find different patterns of proteins between the experimental and control samples, or between the samples from patients with specifics disease and the control samples from healthy patients. The pursuit of new drugs and recent technological advances on large-scale studies of proteins will continue to be a major driver in the biotechnology and health industries.

1.5 Contributions of this Thesis

1. The first contribution concerns the popular and simple FS strategy: correlation-based feature selection (CFS). In experiments reported in this thesis, we see that CFS is never the best choice for poorly correlated data, which means that it may do more harm than good. Although the result is straightforward, the claim can be made that this is a significant contribution, because CFS is often employed, without consideration, to select features in order to reduce the size of vast datasets. In this method, features are chosen that have the best correlation with the 'target' feature, based on the basic statistical correlation with the target feature. However, we show that the features chosen by CFS will be not ideal in the case when the correlations in the dataset tend to be low, and in fact this can lead to underperformance in

later data mining. In effect, we show that a simple test of the dataset can show whether or not it is sensible to use CFS.

2. The second contribution of the work is the description of Feature Bias (FB) techniques, and their experimental evaluation by application on the various datasets in this thesis. FB attempts to have the best of both worlds; the idea of evaluating each feature is used, so that the features that seem most useful are given more importance in the machine learning process; however, unlike FS methods, FB still uses all features, so all of the important features are available to the machine learning process. We find that FB techniques are generally good in their performance, displaying strong capability on some datasets.

3. The third contribution, in close relation to the second, is the concept of combined FS/FB strategies, where FS is used with some caution, and FB is used on the reduced dataset. The thesis contributes empirical evidence for many such combined strategies, and we see strong performance on certain datasets.

4. The fourth contribution is the claim, justified by some statistical analysis, as follows: simple statistical measures of a dataset can be used to predict the relative performance of certain different FS methods. This is a more general and extended version of the first contribution. Put in another way: by making a simple calculation based on the dataset itself, we can determine a good prediction of whether or not it is sensible to use CFS for FS on that dataset. More generally, the measure calculated from the dataset can help us determine which FS (or FB) method to use, from a small set of FS and FB methods.

Related to the fourth contribution, the thesis offers a simple decision guide for choosing an FS method (from the ones reported in this thesis) on the basis of a simple measure of the dataset, which we call the Dataset Correlation Value (DCV).

1.6 Overview of Whole Thesis

The above issues sketch the scope of the present work. Basically, the nature of the research reported here falls into three areas:

1 Comparison and investigation of various FS approaches on large-scale data

2 Investigation of new/alternative FS and related methods.

3 Exploring the relationship between dataset statistics and the performance of the FS method.

The first area aims at finding advantages and disadvantages of the different approaches and yielding a better understanding of the effects and the differences of the various methods. To compare different FS methods, this was done by comparing the performance of machine learning on the reduced datasets. The method of machine learning we chose to use was to evolve a set of rules to classify the dataset. So, part of this research included brief examination of factors of the evolutionary algorithm such as population size and rate of mutation. The purpose of the second area was to learn from the comparison results, and see if this learning can be used to drive different directions for FS strategies. In this area we developed FB strategies and tested them, and also combined FS/FB strategies, and also combinations of basic FS strategies. But most of what we learned from the first comparison studies was used to drive area three, which involves understanding the relationship between the performance of the FS methods and the datasets themselves.

Chapter 2 begins by presenting the concept of feature selection techniques. Three broad categories of technique are discussed – Complete, Heuristic, and Random. It also reviews two typical types of algorithms - filter and wrapper methods - those that do not involve the machine learning scheme to estimate the worth of features, and those that do. Section 2.4 surveys several feature selection methods. The problem of how to choose a feature selection method is discussed in next section. Then in the next we brief review the concept of evolutionary algorithms, and Section 2.7 presents conclusions.

Chapter 3 focuses on the comparison of five popular feature selection strategies. In the first section 3.1, it gives the basic introduction about the feature selection methods. Then section 3.2 describes the datasets and algorithms used in the experiments and section 3.3 outlines the performance of each method and some notes on preliminary experiments. The conclusions are discussed in section 3.4.

Chapter 4 addresses the investigation of advantages and disadvantages of FS and FB methods. We define feature bias (FB) methods in section 4.1 and provide an overview of basic variant FB methods and the combined methods in section 4.2. More datasets are used in the next experiments for comparing the performance of FS and FB. These datasets are presented in section 4.3. In this section we also divide the Datasets into three groups according to their types and discuss the range of correlation values in each type of dataset. Section 4.4 and section 4.5 gives us the results from experiments and the analysis. And section 4.6 discusses the conclusion.

Chapter 5 explores the relationship between the dataset statistics and the various methods. It begins with the introduction in section 5.1, then surveys the concept of statistical correlation coefficient: Pearson's correlation and Spearman's correlation are in mentioned in section 5.2. In section 5.3, it presents the overview of Spearman's correlation and the details of applications used in our experiments. Then in section 5.4 we apply the Pearson's correlation for proving the significance of the correlation between a basic statistic of the dataset, and the performance of CFS and other methods. In section 5.5 we introduce a simple 'Black box' decision strategy to give a guide towards choosing the appropriate FM (feature management) method when we have calculated basic statistics of the dataset. Section 5.6 presents the brief discussion.

Chapter 6 summarises this thesis and offers future perspectives. It includes four sections: the discussion, and the overview of the whole research, the contributions, and future work.

Chapter 2

Literature Review

In section 2.1 we give a broad overview of feature selection. Then in section 2.2 we describe the three main categories of feature selection method. When feature selection is applied, it is usually because the data needs to be reduced or optimised for input to a machine learning method. There are two main ways in which feature selection and machine learning are combined, and these are discussed in section 2.3. In section 2.4 we survey several popular feature selection methods for large-scale datasets. The SVM and SVM-RFE methods are presented in subsection 2.4.1. Then in subsection 2.4.2 we provide an introduction of Relief methods and variants. Subsection 2.4.3 presents many widely used feature selection methods, some of which are based on basic statistical calculation. In section 2.5 we analyse literature concerned with how to choose feature selection methods. Finally in section 2.6 we briefly review evolutionary algorithms (the learning method used in this thesis to test FS performance). The last section presents conclusions.

2.1 Overview of Feature Selection

Feature selection is often found to be an essential pre-processing step in machine learning, wherein a subset of features is selected from the data for classification or prediction of learning algorithm before applying a machine learning algorithm (Jones S S and Smith L B (1992)). The aim of feature selection is to pre-select a relatively small number of attributes, thus speeding up further processing and (hopefully) eliminating data that have minimal or no discriminatory power (John G H, Kohavi R and Pfleger K (1994)). A feature selection algorithm chooses a subset of cardinality *f* from the original *k* features where $f < k$. The technique tries to find the most important or relevant *f* features, hence reducing the dimensionality of the data with a view to reducing the complexity of the problem. This reduction can lead to an immense speedup in processing time, as well as potentially much better generalization performance.

A typical feature selection method consists of four basic parts (defined by Dash M and Liu H (1997)) as in the following, where each part could be seen as an option to select features:

1) Find a starting point to generate a feature subset.

   The starting point – in some methods this is the complete set of features, and features are gradually removed. In others the starting point is the empty set, and features are gradually added

2) Use a selection method (e.g. complete, heuristic, random)

   The method for guiding changes to the starting feature set.

3) Involve an evaluation strategy

   The evaluation strategy: how a score is calculated to estimate the quality of a set of features.

4) Stopping criterion

In more detail:

1) The popular three starting points are respectively (i) start with no features, (ii) start with all features, or (iii) start with a random subset of features. The option with no features will add attributes and test the resulting set, usually one by one. In this case, the search is said to proceed forward through the search space. In contrast, the search could begin with all the features and successively remove them, and this is called to proceed backward through the search space. Alternatively, it could start with a random subset and move outward and/or backward from this point.

2) The 'complete' method is an exhaustive search over the feature subset space. When the feature space is limited to a very small number, this is a guaranteed way to find

the optimal subset. However, with a initial features, there exist $2^a - 1$ possible non-empty feature subsets, and obviously this is no use in general. The heuristic and random selection methods are far more efficient method to guide the search but they do not guarantee finding the optimal subset. The next Section will describe some feature selection strategies that have been used recently.

3) Evaluation strategy. Will the process of machine learning be independent of the feature selection? How to evaluate the feature subsets becomes a critical differentiating factor. One strategy, denominated the *filter* method (Kohavi R (1995)), operates by filtering the irrelevant features out of the data before machine learning begins. In the progress of *filter*, the machine learning algorithm is independent from the feature selection methods. On the other hand, the method, dubbed the *wrapper* method (Kohavi R and John G H (1996)), which estimates the accuracy of feature subsets by using an induction algorithm with a statistical re-sampling technique, is not independent. Section 2.3 discusses the filter and wrapper approaches in details for the connection between feature selection and machine learning.

4) Stopping criterion. A suitable stopping criterion should avoid that the feature selection process runs unnecessarily long. Depending on different search methods, the stopping criterion will be influenced by the generation procedures and evaluation strategy. Still, there are lots of choices for selecting a stopping criterion, such as a predefined number of features selected, or a maximum number of iterations reached.

2.2 Categories of FS Methods

The feature selection (Liu H and Motoda H (1998)) methods fall into three categories as follows:

1. Complete methods: this includes exhaustive and some non-exhaustive methods. The exhaustive methods are guaranteed to find an optimal subset by generating and checking all possible candidate subsets, however it only applied if the time is not an issue and the size of the whole relevant feature set is small. In some cases there can be more features but we are still guaranteed to find an optimal subset using search strategies such as Branch and Bound.

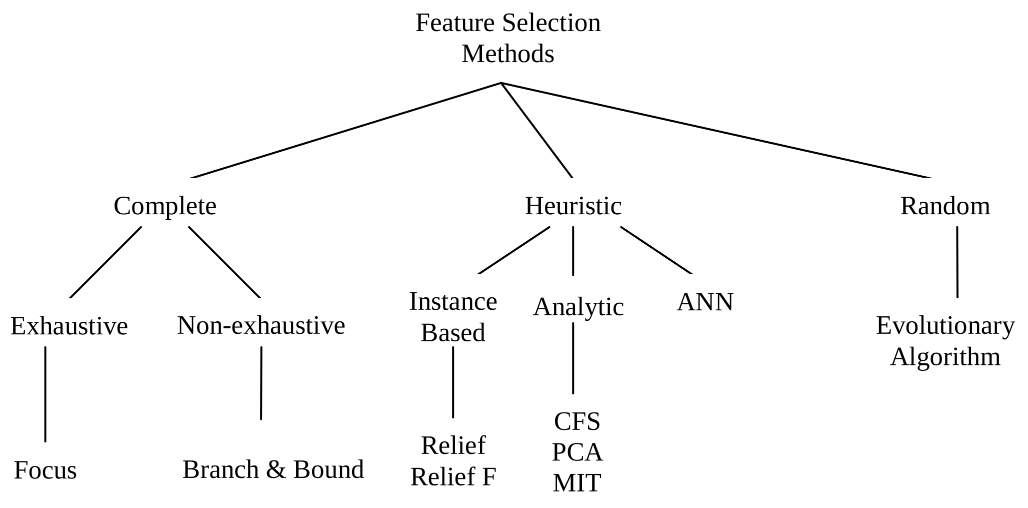


Figure 2.1. A hierarchy of feature selection methods.

A study by Dash M and Liu H (2003) looked at various search strategies. In their research, they looked at five different algorithms: *exhaustive* (Focus), *complete* (ABB), *heuristic* (Set Cover), *probabilistic* (LVF), and a hybrid of complete and probabilistic search methods (QBB). The results could be seen as offering guidelines for a user to select the best algorithm under particular circumstances. Despite the cost of time, the Focus and ABB methods were preferable because they ensured smallest consistent subsets. But in the usual case of limited computing time a user is best guided to choose from LVF and QBB.

Research by Pudil P and Novovicova J (1998) looked to present some guidelines on the method of feature selection to choose based on the knowledge of the problem needing to be solved. A preliminary flowchart was built indicating the methods of feature selection to choose based on the characteristics of the problem. For example, if the total number of features is greater than 30, sequential feature selection methods are recommended otherwise a branch and bound search is suggested.

An optimal feature selection method cannot be improved in terms of accuracy but the time complexity leaves a lot to be desired. An improved branch and bound method, (IBAB) proposed by Chen X (2003), aims to reduce the search time that the conventional branch and bound method usually requires. Partial paths, which are sub paths of branch and bound paths, are searched for. If a partial path is found such that its criterion function value is less than the current stored best for partial paths then all full paths containing this partial path are ignored. However, by reducing the time taken to perform the branch and bound search, optimality is compromised.

2. Heuristic methods: sequential search methods. Although these algorithms may not guarantee minimal size subsets, they will be efficient in generating consistent subsets of size close to minimal in much less time when the number of relevant features and the number of features both are large.

Jain A and Zongker D (1997) evaluate different feature selection methods, looking specifically at their advantages and disadvantages for particular problems. The experiments conducted in the study demonstrated the existence of the curse of dimensionality, also known as Hughes paradox or the peaking phenomenon. For a feature selection algorithm there appears to be an optimal number of features that can be selected. Adding more features causes the classification error to rise. This effect seems counterintuitive. The more information about a problem is used, fewer mistakes should be expected. This effect has been attributed to the fact that traditional Datasets are finite in size and, as such, only imperfect estimates of probability distributions may be found.

Kudo M and Sklansky J (2000) also compare feature selection algorithms for classifiers. The study incorporates a comparison of branch and bound methods, sequential algorithms and genetic algorithms on a variety of small, medium and large Datasets. In conclusion it is seen that the sequential algorithms can give better results than the other methods for the small and medium sized datasets.

Gadat S and Younes L (2007) introduce a new model addressing feature selection from a large dictionary of variables that can be computed from a signal or an image. Features are extracted according to an efficiency criterion, on the basis of specified classification or recognition tasks. This is done by estimating a probability distribution *P* on the complete dictionary, which gives most probability to the more efficient, or informative, components. A stochastic gradient descent algorithm is implemented by using the probability as a state variable and optimizing a goodness of fit criterion for classifiers based on variables randomly chosen according to *P*. Then classifiers are generated from the optimal distribution of weights learned on the training set. Several pattern recognition problems including face detection, handwritten digit recognition, spam classification and micro-array analysis, are tested for this experiment. The results show that the performance is significantly improved over an initial rule in which features are simply uniformly distributed. Optimal Feature Weighting method (Scherf M and Brauer W (1997)) is moreover competitive in comparison with other feature selection algorithms and leads to an algorithm which does not depend on the nature of the classifier which is used, whereas, for instance, RFE or L0-SVM are only based on SVM.

3. Random methods: These methods generate the candidate subsets randomly but often use a supervised guidance which allows mutation in the logic for searching alternative areas of the feature space. This random method cannot guarantee the discovery of the optimal subset.

The research by Juliusdottir T *et al.* (2005) investigates a simple evolutionary algorithm/classifier combination on two microarray cancer datasets, where this combination is applied twice – once for feature selection, and once for further selection and classification. Their contribution are: (further) demonstration that a *simple* EA/classifier combination is capable of good feature discovery and classification performance *with no initial dimensionality reduction*; demonstration that a simple *repeated* EA/*k*-NN approach is capable of competitive or better performance than methods using more sophisticated pre-processing and classifier methods.

Even though the complete methods guarantee the expectation of the optimal candidate subset, it costs a high price to implement such methods which require high computational complexity. For this reason, and especially with the size of datasets today in the bioinformatics field, heuristic and random methods are getting more considered and widely implemented frequently in spite of not having the guarantee of optimality.

2.3 Filter and Wrapper

There are two broadly different ways in which FS strategies are applied in combination with machine learning. These are the *Filter* and *Wrapper* methods. The filter method, defined by Kohavi R and John G H (1996), which evaluate the worth of features based on general characteristics of the data, is a pre-processing step, independent of the choice of the machine learning method. And the wrapper methods, those which evaluate the worth of features by using the special learning algorithm that is to ultimately be implemented to the data is to assess subsets of variables according to their usefulness to a given predictor. Within both categories, algorithms can be further differentiated by the exact nature of their evaluation function, and by how the space of feature subsets is explored.

2.3.1 Feature Filter Approach

The filter method is the earliest approach to feature selection. It uses the intrinsic properties of the training set to decide which features to reserve or to discard. As the filter methods utilise an indirect measure to find the appropriate feature subset, it is often done a priori before an induction algorithm is performed. It filters out redundant or irrelevant attributes before machine learning occurs, that is the search is done independently of the machine learning algorithm. The advantage of the filter model is that it does not need to re-run the algorithm for every induction algorithm when choosing to run on a reduced feature dataset, as a consequence, the filter approach is generally computational efficient, and it is practical for datasets with very high dimensionality.

The filter selection procedure is illustrated in Figure 2.2. As shown in this procedure, the Filter model includes three steps. The first step is to select the filter algorithm as an operator to guide the search. Usually, the performance of the filter algorithm directly results in the selection of variable subsets. Many filter operators are deterministic (such as CFS), which means that, for a given dataset, they will always rank the features in the same way. In other cases, such as using evolutionary algorithm feature selection, the ranking of the features can be different when applied different times to the same data.

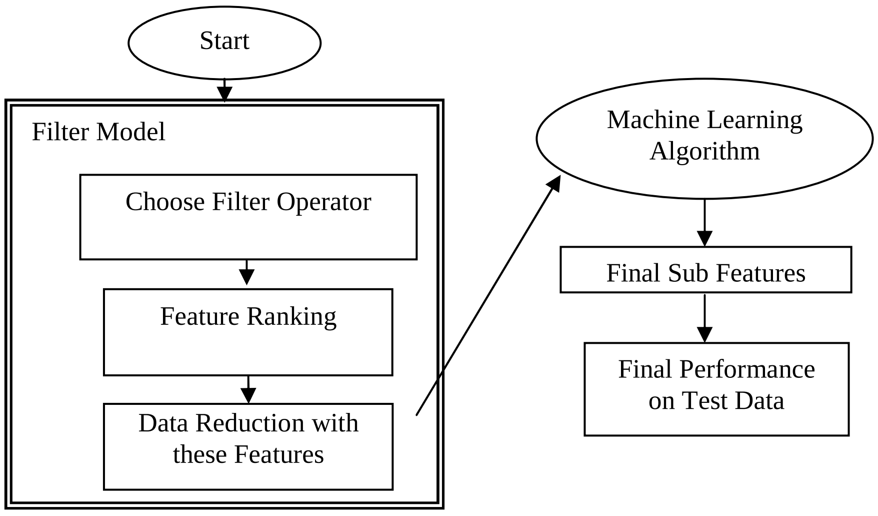

Start
Filter Model
Choose Filter Operator
Feature Ranking
Data Reduction with these Features
Machine Learning Algorithm
Final Sub Features
Final Performance on Test Data

Figure 2.2. The filter Selection procedure.

The second step is ranking all the features to choose the most fit *n* features. In most high dimensional data (e.g. with thousands of attributes), few hundred features could lead to better performance in the further work; however, in some (usually smaller) datasets, around 10 features may produce better performance. The third step is to compress the original dataset by removing the features that were not selected. The methods studied in this thesis are all filter style methods. The details will be discussed in chapter 3 and chapter 4.

There are a number of different commonly used filter algorithms such as CFS and Relief. These representative filter approaches works as follows: they first evaluate the individual features according to an evaluation criterion, and there afterwards, the best *n* features are selected, then the resulting subset of features is then fed as input to the machine learning system. This class of techniques essentially produces a feature weighting scheme, which is used afterwards to rank and select features.

Kira K and Rendell L A (1992) introduced the Relief algorithm which follows this general paradigm but incorporates a complex feature-evaluation function. In their work, they then use ID3 (Quinlan J R (1983)) to induce a decision tree from the training data using only the selected features. Kononenko I (1994) reports two extensions to this method that handle more general types of features. The next subsection will describe more details of the Relief algorithm.

Yu L and Liu H (2003) introduce a novel concept, predominant correlation, and propose a fast filter method which can identify relevant features as well as redundancy among relevant features without pairwise correlation analysis. The efficiency and effectiveness of their method is demonstrated through extensive comparisons with other methods using real-world data of high dimensionality.

Since the filter approach does not take into account the learning bias introduced by the final induction algorithm, it may not be able to select the most suitable subset for the final induction algorithm. For this reason, the wrapper model was proposed.

2.3.2 Feature Wrapper Approach

The strategy of the wrapper model is to use an induction algorithm to estimate the merit of the searched feature subset on the training data and using the estimated accuracy of the resulting classifier as its metric. The rationale for wrapper approaches is that the induction method that will ultimately use the feature subset should provide a better estimate of accuracy than a separate measure that has an entirely different inductive bias Langley P (1994).

As exhibited in the Figure 2.3, machine learning plays an important role in the Wrapper approach. The wrapper approaches often have better results than the filter approaches because they are tuned to the specific interaction between an induction algorithm and its training data. The disadvantage of the wrapper model is that it is less tractable because of the prohibitive cost of running the classification algorithm many times when the dimensionality is considerably high. As a result of the wrapper methods must repeatedly call the induction algorithm and must be re-run when a different induction algorithm is used, they tend to be much slower than filter methods. So, the wrapper approach is not our method of choice in this thesis as we look into large-scale datasets.

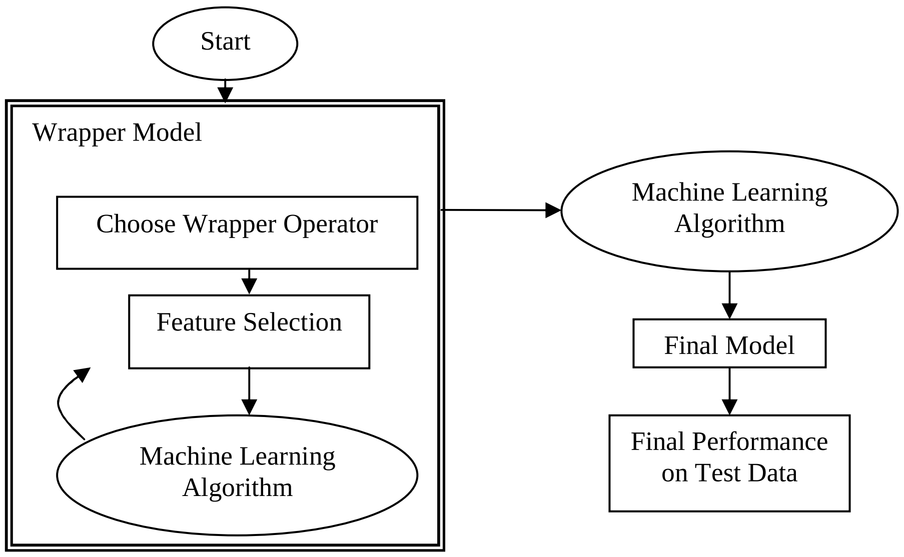

Start
Wrapper Model
Choose Wrapper Operator
Feature Selection
Machine Learning Algorithm
Machine Learning Algorithm
Final Model
Final Performance on Test Data

Figure 2.3. The procedure of wrapper selection.

For instance, Aha D W and Bankert R L (1996) report a technique which starts with a randomly selected subset of features and includes an option for beam search rather than greedy decisions. They report impressive improvements on a cloud classification task that involves over 200 numeric features. Skalak's D B (1994) work on feature selection for nearest neighbour is a wrapper approach that also starts with a random feature set, but replaces greedy search with random hill climbing that continues for a specified number of cycles.

Finally this subsection has basically introduced the concepts of filter and wrapper methods. Both of them could find potential best results in finite computation time. However, the filter method has an advantage in dealing with large many-attribute datasets. In the next chapter, we will describe more details about some other widely-used feature selection methods on high dimensional datasets. Some of them could be applied in both filter or wrapper modes, such as SVM and EA, however, some of them are usually only used as filter, for example, CFS and Relief.

2.4 Feature Selection Methods on Large-scale Dataset

2.4.1 SVM and SVM-RFE

SVM

The state-of-the-art classification algorithms, Support vector machines (SVM) by Boser B E (1992) are a group of related supervised learning methods that can be applied to classification or regression. These have been demonstrated to be an effective tool when used for a variety of applications in object recognition to classification of cancer morphologies and a variety of other areas.

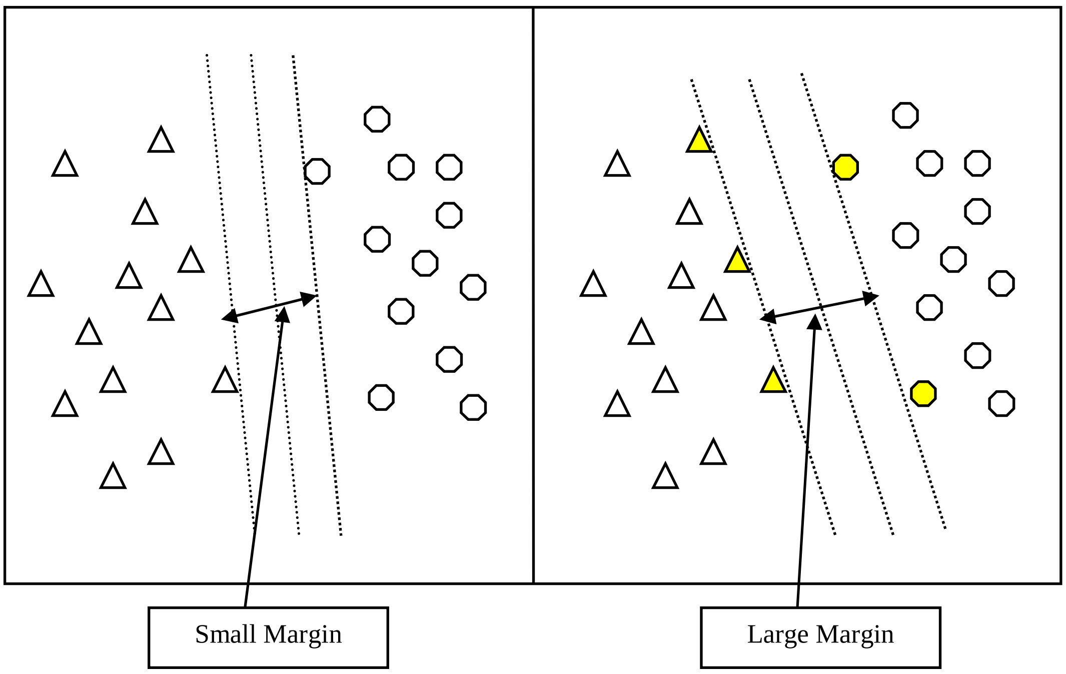


Figure 2.4. It illustrates the operation of SVM Classifier. Yellow ones are support vectors.

In linear SVM classification, a hyperplane with maximum margin means are constructed to linearly separate two classes. For an instance as shown in Figure 2.4, the data points are two sets of vectors, the round ones and the square ones, and the question is how to find out the optimal hyperplane that separates these two classes. The distance between the dashed lines is called the *margin*. The vectors (points) that constrain the width of the margin are the *support vectors*. When the margin between the support vectors is maximised, this oriented line in the right of Figure 2.4 is the better classifier than one in the left for this classification problem.

As there exists many problems that have no hyperplane that can split two classes completely, a modified maximum margin idea, allowing for mislabelled examples, was presented by Cortes C and Vapnik V (1995). This soft margin method defined a misclassification error ξ_i "slack variables", and employs a parameter C to control the cost of misclassification.

Viewing a typical classification problem with a training set of instance-label pairs (x_i y_i) i = 1…m, where $x_i \in R^n$ and $y_i \in \{1, -1\}^m$, the support vector machines attempt to find the discriminant function $f(x) = w \cdot x_i + b$ which is formulated by SVMs into the following optimization problem.

Minimising $$\tfrac{1}{2}\,||\mathrm{w}|| + C\sum_{i=1}^{m}\xi_i \qquad (1)$$

Under the constraints:

$$y_i[(w \cdot x_i)] + b \geq 1 - \xi_i \text{ and } \xi_i \geq 0 \qquad (2)$$

where C is a tradeoff parameter between error and margin. If ξ_i =0, there is no error for x_i . At this condition, the optimization problem is min $\frac{1}{2}$ ||w|| subject to $y_i[(w \cdot x_i)] + b \geq 1$, which is the formalization of original linear optimal hyperplane algorithm .

This optimization problem is usually solved in its dual form:

$$\max \sum_{i=1}^{m} a_i - \frac{1}{2}\sum_{i,j} a_i a_j y_i y_j (x_i \cdot x_j) \qquad (3)$$

$$\text{subject to } y_i \geq 0 \text{ and } \sum_{i=1}^{m} a_i y_i = 0 \qquad (4)$$

where w is recovered by using the dual variables as a_i : w = $\sum_{i=1}^{m} a_i y_i x_i$ (5)

So far, we have only considered large-margin classifiers with a linear decision boundary (the hyperplane). The study by Boser B E, Guyon I and Vapnik V (1992), applied the ‘kernel trick’ to create non-linear classifiers. The mapping ϕ, which represent the data from the input space to feature space, is usually nonlinear and the feature space is a much higher (possibly infinite) dimensional space than the original input space. The kernel function computes the inner product of two vectors in the feature space and implicitly defines the mapping function:

$$k(x_i, x_j) = \phi_{(x_i)}\, \phi_{(x_j)} = (x_i \cdot x_j) \qquad \text{Linear Kernel} \quad (6)$$

There are other commonly used kernel functions as following: Polynomial Kernel, Gaussian, Kernel, radial basis function, sigmoid. If the non-linear kernel function is used, the above optimization problems will change correspondingly. In these cases, because of the nonlinear mapping relation between the input space and the feature space, the linear discriminate function constructed by an SVM in the feature space corresponds to a nonlinear function in the original input space. Therefore the classification could grow easier with a proper transformation.

SVMs (Burges C J C (1998)) provide a new approach to the problem of pattern recognition with clear connections to statistical learning theory. They differ radically from comparable approaches such as neural networks: SVM training always finds a global minimum for separating the classes, and their simple geometric interpretation provides lots of ideas for further investigation. An SVM is largely characterised by the choice of its kernel, and SVMs thus link the problems they are designed for with a large body of existing work on kernel based methods.

Jong K, *et al.* (2004) proposed two combination strategies: union of features occurring frequently, and ensemble of classifiers built on single feature subsets. The resulting methods are applied to pattern proteomic data for tumor diagnostics. Results of experiments on three proteomic pattern datasets indicate that combining feature subsets affects positively the prediction accuracy of both SVM and SVM-RFE. It suggests SVM has been used throughout this investigation for feature ranking/selection and for classification. However, JOIN and ENSEMBLE can be applied to feature subsets produced by any other method.

Chen Y W and Lin C J (2006) investigate the performance of combining support vector machines (SVM) and various feature selection strategies. Some of them are filter- type approaches: general feature selection methods independent of SVM, and some are wrapper-type methods: modifications of SVM which can be used to select features. The experience indicates that for such problems, SVM can handle a rather large set of features.

SVM-RFE

The Support Vector Machine Recursive Feature Elimination (SVM-RFE) method was originally proposed to perform gene selection for cancer classification by Guyon I (2002). This idea of SVM -RFE could be used as a filter or wrapper approach based on backward sequential selection. The method is to start with all features, select the "least useful" feature (ranking features by a linear SVM on the training set), and remove that feature. In this way, a chain of feature subsets of decreasing size is obtained. At each interaction, the coefficients of the weight vector w of a linear SVM are used as the feature ranking, and more than one feature is discarded for speed reasons.

The flowchart of the SVM-RFE method was shown as follows:

```
1)  Initialization:
        Ranked feature $R$ = []; Feature ranking list $r$ = [] ;
        Subset of survival features $S$ = [1…m];
2)  Repeat until all feature are ranked S = [] :
    a)  Train the SVM classifier with the training data : $a = SVM - Train(x, y)$;
    b)  Compute the weights $w = \sum_{i=1}^{m} a_i y_i x_i$;
    c)  Compute the ranking scores for features in $S$: $c_i = (w_i)^2$;
    d)  Find the feature with smallest ranking criterion : $f = argmin(c)$;
    e)  Remove the variable $f$ : $S$ =[1,…, $f$−1 , $f$+1,…,$m$] and update $r$ = [ $S(f)$ , $r$] ;
3)  Output: Feature ranked list $r$.
```

Ranking criterion c_i could be chosen differently to rank variables. Using $(w_i)^2$ as the ranking score corresponds to removing the feature whose removal changes the objective function least. However, many different criterions are used frequently to compare the performance. Weston J, *et al.* (2001) used the radius/margin bound for feature selection using a gradient descent algorithm.

Two approaches by using c_i differently, zero-order method and first-order method could be explained as respectively the filter or wrapper approach. In the case of the zero–order method, the criterion c_i is directly used for variable ranking, and the method consists in identifying the variable that produces the smallest value of $f(c)$ when removed. The first-order method uses the derivatives of the criterion $f(c)$ with regards to a variable. In other words, this approach differs from the previous one since a variable is ranked according to its influence on the criterion which is measured with the absolute value of the derivative.

A study by Duan K and Rajapakse J C (2005) compare the performance of a linear SVM-RFE and a basic T-statistics method on two cancer classification mass spectrometry datasets: this demonstrated that SVM-RFE can select a good small subset of peaks with which the classifier achieves high prediction accuracy and the performance is much better than with the feature subset selected by T-statistics. And some features selected from SVM-RFE are ranked top by the T-statistics as well.

Mao Y, Zhou X, Pi D, Sun Y, and Wong T C (2005), introduce two extensions of SVM-RFE: a binary classification tree based on SVM (BCT-SVM) and FSVM with SVM-RFE. FSVM is an improved pairwise classification method to deal with unclassifiable regions. Binary classification tree SVM is to build a binary tree by searching with SVM at each internal node, to find where best to separate the data in the current node into two children nodes with appointed gene selection method.
Another group Tang Y, Zhang Y, Huang Z and Hu X (2005) from Georgia State University, presents the novel Granular Support Vector Machines-Recursive Feature Elimination (GSVM-RFE) algorithm for the gene selection task. GSVM-RFE can separately eliminate irrelevant, redundant or noisy genes in different granules at different stages and can select positively related genes and negatively related genes in

balance. In their research, GSVM-RFE extracts a compact “perfect” gene subset of 17 genes with 100% accuracy on prostate cancer dataset.

2.4.2 Instance–based Learning Algorithms

Instance-Based Learning (IBL) is defined as the generalizing of a new instance (target) to be classified from the stored training examples. Generalizing beyond these examples is postponed until a new instance is classified. Sometimes these are called *Lazy Learning*. Each time a new instance is encountered, its relationship to the previously stored examples is examined in order to assign a target function value for the new instance.

Some techniques only construct a local approximation of the target function that applies in the neighbourhood of the new query instance and they never construct an approximation designed to perform well over the entire instance space. This has a significant advantage when the target function is very complex, but can still be described by a collection of less complex local approximations. The disadvantages of instance–based approaches are that the cost of classifying new instances can be high. This is based on the fact that nearly all computation takes place at classification time rather than when the training examples are first encountered. Therefore, techniques for efficiently indexing training examples are a significant practical issue to reduce the computation required at query time. A second disadvantage to many instance-based approaches, especially the commonly used nearest neighbour approach, is that they typically consider all attributes of the instances when attempting to retrieve similar training examples from the memory. If the classification of the target instance depends on only a few of the many available attributes, then the instances that are truly most “similar” may well be a large distance apart.

Aha D W (1992) defines the framework of instance-based learning algorithms that have three components:

- A similarity function:
  Compute the similarity between a training instance and the instance in the concept description. This function calculated the distance that means how

close together two instances are. Similarity function plays an important role, especially in situations where some of the inputs are enumerated. For example, if you were trying to distinguish an image of a human being, and one attribute was shape of face, a proper similarity function could influence the distance between two instances.

- A Concept description updater:
  This maintains information on classification performance and chooses which instances to include in the concept description. It determines which of the instances to keep as examples. The modified concept description will take place the previous one.

- A classification function:
  This uses the similarity functions result and the classification performance records of the instances in the concept description to yield a classification.

IB1 IB2 IB3

There are many variations on the basic theme of IB. There are three that Aha D W, Kibler D and Albert M K (1991) propose in their paper: *IB1*: Store all example instances and simply find the closest instance -- then the class of this instance is the class of the closest instance. The large number of instances that need to be stored, however, can require a large amount of space. *IB2*: Because of throwing away instances in the training set that would have already have been correctly classified, the storage requirements could be significantly smaller than IB1 where the instances vary greatly in their distance from the concept boundary. *IB3*: is a noise –tolerant extension of IB2 wherein the noisy instances are almost always misclassified. This version makes some assumptions about the data and uses a statistical methods to “weed out” irrelevant or noisy instances. Not only does IB3 reduce the size of IB1 and IB2 storage requirements, it also shows reduced sensitivity to noise.

The most common IBL methods are: 1) *k*-Nearest Neighbour; 2) Locally Weighted Regression; 3) Radial Basis Functions. The next section will describe the basic concept of k-nearest neighbour learning algorithm. The *locally weighted regression* methods are a generalization of k-nearest neighbour in which an explicit local approximation to the target function is constructed for each instance that needs to be classified. The local approximation to the target function may be based on a variety of functional forms such as constant, linear, or quadratic functions or on spatially localised kernel functions. *RBF*, radial basis functions, is a type of artificial neural network constructed from instance–based approaches (spatially localised kernel functions). The artificial network could be seen as a global approximation to the target functions that is formed at training time. Therefore the RBF is a combination of both and have been used successfully in applications such as interpreting visual scenes.

K-Nearest Neighbour

The *k*-Nearest Neighbour algorithm is the most basic Instance-Based Learning (IBL) method for approximating real-valued or discrete-valued target function. The algorithm assumes all instances correspond to points in the *n*-dimensional Euclidean space R_n (Baily T and Jain A K (1978)). Let an arbitrary instance $\boldsymbol{x}$ be described by the feature attribute lists: < $a_1(x)$, $a_2(x)$, $a_3(x)$, ..., $a_n(x)$>, where $a_r(x)$ denotes the value of the $\boldsymbol{r}$ th attribute of instance $\boldsymbol{x}$ The distance between the two instances x_i and x_j is given by:

$$d(x_i, x_j) \equiv \sqrt{\sum_{r=1}^{r=n} \left[a_r(x_i) - a_r(x_j)\right]^2}$$

This is the general form for calculating distance in *n*-dimensional space.

In nearest-neighbour learning, the target function may be either discrete-valued or real-valued. Consider learning discrete-valued target functions of the form $\boldsymbol{f}$**:** $\boldsymbol{R^n}$**->**$\boldsymbol{V}$, where $V = \{v_1, v_2, v_3, ..., v_s\}$ is a finite set and $\boldsymbol{R_n}$ is real *n*-dimensional space. The *k*-Nearest Neighbour algorithm for approximating a discrete-valued target function is given below. This algorithm illustrates the operation of the *k*-nearest neighbour algorithm for the case where the instances are points in a two–dimensional space and wherein the target function is Boolean valued.

```
Training Algorithm:

For each example <x, f(x)>, add the example to the list training examples
Classification Algorithm:
Given a query instance $x_q$ to be classified

    Step 1: Let $x_1$, $x_2$, ..., $x_k$ denote the $k$ instances from the training examples that are nearest to $x_q$.

    Step 2: Return: $f(x_q) \leftarrow \arg\max_{v \in V} \sum_{i=1}^{i=k} \delta(v, f(x_i))$

    Where $\delta(a,b) = \begin{cases} 1 & if\,(a = b) \\ 0 & if\,(a \neq b) \end{cases}$ ( argmax means maximum of function ).
```

Figure 2.5. The k-Nearest Neighbour algorithm for discrete-valued function.

The algorithm for continuous-valued target functions is the same as shown above, except that *Step 2* is replaced with the following expression:

$$f(x_q) \leftarrow \frac{\sum_{i=1}^{i=k} f(x_i)}{k}$$

Relief, Relief-F

Relief is considered as one of the most successful feature weighting algorithms Dietterich T G (1997). It is often applied in a prepossessing step before the model is learned. Relief is an instance based learning to assign a relevance weight to each feature (Kira K and Rendell L A (1992a)). Due to its simplicity and effectiveness, it has been used successfully in a variety of settings: to select splits in the building phase of decision tree learning (Kononenko I, Simec E, and Robnik S M (1997)), to select splits and guide the constructive induction in learning of the regression trees (Robnik S M and Kononenko I (1997)), as attribute weighting method **(**Wettschereck D, Aha D W, and Mohri T (1997)) and also in inductive logic programming (Pompe U and Kononenko I (1995)).

Relief—Basic Ideas

The basic idea, given by (Kira K and Rendell L A (1992b)), is to measure the relevance of features in the neighbourhoods around target samples. The weights of features would be iteratively estimated according to their ability to discriminate between neighbouring patterns. For each target sample, given a randomly selected example *Ri*, Relief searches for the nearest sample in feature space of the same category, called the "nearest hit" sample Rp. It also finds the nearest sample of the other category, called the "nearest miss" sample Rn. Then it updates the quality estimation w_i for all attributes *i* depending on their values for *Ri*, *Rp*, and *Rn* (lines a and b).

The procedure of algorithm Relief is described as following:

Input: for each training instance a vector of attribute values and the class value

Output: the vector W of estimations of the qualities of attributes

```
1. Set all feature weights $w_i$ to 0.
2. Repeat the following for m iterations
    a. Choose a random data record Ri.
    b. Find a sample Rp and Rn, respectively the one record closest (in Euclidean) to Ri which
       are in the positive class, and the one record closest to Ri in the negative class.
    c. For each field i set :
           $w_i = w_i - diff(i, Ri, Rp)/m + diff(i, Ri, Rn)/m$
       Where $diff(i, Ri, Rp)$ indicates the absolute difference between the value of attribute i
       in record Ri, and in record Rp, as a proportion of the range of values in the dataset for
       attribute I. For numerical attribute: $diff(i, Ri, Rp) = \frac{|value(i, Ri) - value(i, Rp)|}{\max(i) - \min(i)}$
```

Figure 2.6. The procedure of basic Relief algorithm.

The short point of the original Relief method is that it cannot deal with incomplete data and is limited to two-class problems. Its extension, which solves these and other problems, is called Relief-F.

The work by Bins J and Draper B A (2001), addresses the feature selection problem by proposing a three-step algorithm. The first step uses a variation of the well known Relief algorithm to remove irrelevance; the second step clusters features using K-means to remove redundancy; and the third step is a standard feature selection algorithm. This three-step combination is shown to be more effective than standard feature selection algorithms for large datasets with lots of irrelevant and redundant features. It is also shown to be no worse than standard techniques for datasets that do not have these properties.

Relief-F

A heuristic algorithm, called Relief-F (Kononenko I (1994)), has been proposed to deal with incomplete and noisy data. Relief-F averages *k*, instead of just one, nearest neighbours in computing the sample margins. Therefore, it is more robust and applicable than *Relief*. Empirical studies have shown that Relief-F can achieve significant performance improvement over the original Relief. User-defined parameter *k* controls the locality of the estimates. For most purposes it can be safely set to 10 (see Kononenko I (1994)) Multiclass datasets are handled by finding the nearest neighbours from each class that are different from the current sampled instance, and weighting their contributions by the prior probability of each class estimated form the training data. The feature weights computed by Relief-F minimise the expected distance between a sample and the *k* nearest samples of the same class, while maximizing the expected distance between a sample and the *k* nearest samples of the other class. The update formula is similar to that of Relief, except that we average the contribution of all the hits and all the misses. The contribution for each class of the misses is weighted with the prior probability of that class $P(C)$ (estimated from the training set).

$$w_i = w_i - (1/k \times m)\sum_{j=1}^{k} d(i, Ri, Rp_j)/(k \times m)$$
$$+ \sum_{C \neq class(Ri)} \frac{P(C)}{1 - P(Ri)} \sum_{j=1}^{k} d(i, Ri, Rn(C)_j)/(k \times m)$$

Where *Rn* (*C*) stands for the nearest miss samples with class *C*.

To deal with incomplete data we change the *diff* function. Missing values of attributes are treated probabilistically. We calculate the probability that two given instances have different values for given attribute conditioned over class value.

If one instance (e.g. *Ri*) has unknown value:

$$diff(i, Ri, Rp) = 1 - P(value(i, Rp)/value(Ri))$$

If both instances have unknown value:

$$diff(i, Ri, Rp) = 1 - \sum_{V}^{values(i)} \left(P(V/class(Ri)) \times P(V/class(Rp))\right)$$

A new feature weighting algorithm, was proposed by Sun Y and Li J (2006). All stemming are from a new interpretation of Relief as an online algorithm that solves a convex optimization problem with a margin-based objective function. The new interpretation explains the simplicity and effectiveness of Relief-F, and enables us to identify some of its weaknesses. Some experiments based on the UCI and microarray datasets are performed to demonstrate the effectiveness of this proposed algorithm. For classification purposes, some computationally expensive methods (e.g. wrapper methods) can be used to further filter out some redundant genes. By using some sophisticated classification algorithms such as SVM, much improvement on classification performance is expected.

2.4.3 Other Methods

T-statistics and MIT

T-statistics, which basically means the ratio of the coefficient of correlation to its standard error, is a filter based feature selection method (Brewer J K (1985)). It selects the feature variables that are most relevant to the concept under study. A ranking score is computed for each feature. It uses the following feature ranking criterion.

$$T(x_j) = \frac{\left|u_j^+ - u_j^-\right|}{\sqrt{\frac{(\delta_j^+)^2}{n_+} + \frac{(\delta_j^-)^2}{n_-}}}$$

where u_j^+ and u_j^- are the mean values of the i^{th} feature respectively over positive and negative samples; δ_j^+ and δ_j^- are the corresponding standard deviations; n + and n − denote the number of positive and negative training samples. This T-statistics function fundamentally measures the normalised feature value difference between two groups. When making selection, we simply take those features with the highest scores as the most discriminatory features.

MIT correlation is also known as signal-to-noise statistics. The score for each feature can be calculated by a slightly different formula as shown below:

$$T(x_j) = \frac{\left|u_j^+ - u_j^-\right|}{\delta_j^+ + \delta_j^-}$$

Correlation-Based Feature Selection Methods

Correlation-Based Feature Selection Methods are algorithms that couples this evaluation formula with an appropriate correlation measure and a heuristic search strategy (Hall M A (1998)). The correlation value, a single number that describes the degree of relationship between two variables, is one of the most common and most useful statistics. In other words, a feature is useful if it is correlated with or predictive of the target class; otherwise it is considered to be irrelevant. The correlation values for each variable in the training data are calculated, and, usually, a predetermined number of features with the highest correlations are chosen. The usual measure used is as follows:

$$c_{f,p} = \frac{n\sum_{i=1}^{n} f_i p_i - \left(\sum_{i=1}^{n} f_i\right)\left(\sum_{i=1}^{n} p_i\right)}{\sqrt{\left(n\sum_{i=1}^{n} f_i^2 - \left(n\sum_{i=1}^{n} f_i\right)^2\right)\left(n\sum_{i=1}^{n} p_i^2 - \left(n\sum_{i=1}^{n} p_i\right)^2\right)}}$$

where $c_{f,p}$ is the correlation between feature *f* and the target feature *p*, *n* is the number of (*f*, *p*) value pairs in the data, while f_i is the value of feature *f* in the *i*th such pair. The value of $c_{f,p}$ will always be between −1.0 and +1.0. If the correlation is negative, we use the absolute value of it to compare with other values. A good feature subset is one that contains features highly correlated (predictive of) the class, yet uncorrelated with (not predictive of) each other. In the next chapters will compare CFS with other algorithms in detail.

The use of feature generation and feature selection for TIS (Translation Initiation Sites) prediction was explored by Zeng F, *et al.* (2002). In this paper, the authors combined the use of correlation based feature selection (CFS) with a wide range of classifiers and combinations of classifiers. Their results are achieved with only a small subset of features.

In the research by Yu L and Liu H (2003), they introduce a novel concept, predominant correlation, and propose a fast filter method which can identify relevant features as well as redundancy among relevant features without pair-wise correlation analysis. The feature selection results are further verified by applying two different classification algorithms to data, with and without feature selection. Their application widely extends the correlation based feature selection method on high dimensional data, and smoothly handles data with different types.

The χ^2-Statistics

Since its introduction in 1900 by Karl Pearson, the chi-square (χ^2) test has become the most widely used measure of the significance to which experimental results support or refute a hypothesis. Applicable to any experiment where discrete results can be measured, it is used in almost every field of science. It is a nonparametric statistical technique using frequencies instead of means and variances. Generally the *chi-squared statistic* summarises the discrepancies between the expected number of times each outcome occurs (assuming that the model is true) and the observed number of times each outcome occurs, by summing the squares of the discrepancies, normalised by the expected numbers, over all the categories (Dorak M T (2006)).

Mathematically, we can calculate the chi-square statistic χ^2 by

$$\chi^2 = \sum (O - E)^2 / E$$

where χ^2 is the chi-square statistic, *O* is the observed frequency and *E* is the expected frequency. The χ^2 method evaluates features individually by measuring their chi-squared statistic with respect to the classes. After calculating the χ^2 value of all considered features, these values would be sorted with the largest one at the first position, as the larger the χ^2 value, the more important the feature is, in terms of the amount of information that feature contains about the classification.

There are two types of chi-square test: 1) The Chi-square test for goodness of fit; 2) The Chi-square test for independence. The Chi-square test for goodness of fit compares the expected and observed values to determine how well an experimenter's predictions fit the data; the chi-square test for independence is used to determine the relationship between two variables of a sample. In this context independence means that the two factors are not related. Typically in social science research, it is interested in finding factors which are related.

Principal Components Analysis (PCA)

PCA (Anton H (2005)) involves a mathematical procedure which is widely used to analyse the relationship between the individual points in a large set of data. It is a common statistics technique for identifying patterns in data of high dimension, and expressing the data in such a way as to highlight their similarities and differences. PCA is applied abundantly in all forms of analysis from neuroscience to computer graphics such as face recognition and image compression. Because it is a simple, non-parametric method of extracting relevant information from confusing datasets. The main advantage of PCA is to identify new meaningful underlying variables so that the data could be compressed by reducing the dimensionality of the dataset without much loss of information. Sometimes it reveals the hidden, simplified structure that often underlies the original data.

Method description: Assuming there is an *n*-dimensions in the data, $\{x_1, x_2, \dots, x_n\}$:

Step 1: Subtract the mean.

Calculate the mean value (the average) of each dimension in the data, then subtract the mean from each of the data dimensions. For each dimension, each of the values x_n is subtracted from the mean $\bar{x}_n$ (the mean of the values of all the data points). This produces a dataset whose mean is zero.

Step 2: Calculate the covariance matrix.

$$Cov(X_{n*n}) = \begin{pmatrix} \operatorname{cov}(x_1, x_1), \operatorname{cov}(x_1, x_2), \dots, \operatorname{cov}(x_1, x_n) \\ \operatorname{cov}(x_2, x_1), \operatorname{cov}(x_2, x_2), \dots, \operatorname{cov}(x_2, x_n) \\ \dots \\ \operatorname{cov}(x_n, x_1), \operatorname{cov}(x_n, x_2), \dots, \operatorname{cov}(x_n, x_n) \end{pmatrix}$$

Where $\operatorname{cov}(x_i, x_j)$ is the covariance value calculated between the *i*th dimension and the *j*th dimension by using the function:

$$\operatorname{cov}(x_i, x_j) = \frac{\sum (x_i - \bar{x}_i)(x_j - \bar{x}_j)}{n-1}$$

Step 3: Calculate the eigenvectors and eigenvalues of the covariance matrix.

Since the covariance matrix is square, calculate the eigenvectors and the corresponding eigenvalues. The eigenvectors provide us with information about the patterns in the data, and the eigenvector with the *highest* eigenvalue is the *principal component* of the dataset. Further information about eigenvectors in general, how to find them, and related matters, can be found in many places, including the textbook (Anton H (2005)).

Step 4: Choosing components and forming a feature vector.

Once the eigenvectors are found from the covariance matrix, sort the eigenvectors by eigenvalue from highest to lowest. This shows the significance of the components. A *feature vector*, is constructed by taking the eigenvectors that are chosen from the list of eigenvectors, then form a matrix with these eigenvectors in the columns. It will lose some information since we *ignore* the components of less significance, However the missing information could be trivial if the eigenvalues are small numbers. While choosing the first *k* eigenvectors, the final dataset has reduced to only *k* dimensions from its original *n*-dimensions.

$$\text{Feature vector} = \left(eig_1, eig_{2,} \ldots, eig_k\right)$$

Step 5: Deriving the new dataset.

To derive the new dataset, simply multiply the feature vector we selected with *k* dimensions in the last step, and the transposed mean-adjusted data.

A note on the general performance of basic statistical FS methods

An interesting competition in feature selection was proposed by Guyon I, Gunn S, Hur A B and Dror G (2004). Five datasets were gathered from different application domains and called for classification results using a minimal number of features. The competition took place over a period of 13 weeks and attracted 78 research groups. The challenge demonstrated both that feature selection can be performed effectively and that eliminating meaningless features is not critical to achieve good classification performance. A filter as simple as the Pearson correlation coefficient proves to be very effective, even though it does not remove feature redundancy and therefore yields unnecessarily large feature subsets. Principal Component Analysis was successfully used by several researchers to reduce the dimension of input space to a few hundred features, without any knowledge of the class labels. It is surprising that some of the best entries used all the features.

2.5 How to Choose a Feature Selection Method

In this chapter, we have introduced many popular methods which have already been proved as successful feature selection algorithms on different datasets. As there is no single method which could be seen as the best solution to various practical difficulties, a problem that seems worth considering is that of how to guide a user towards the right choice of feature selection method for their data. This can be a very important choice, since the wrong choice may lead to important features being removed. This can have major consequences. For example, the application may be to find an accurate diagnostic test based on proteomics data, and wrong choice of features might lead to diagnostic tests that will have more true negatives and false positives than we want.

Independently of the quality of the machine learning methods, or the way in which the machine learning is combined with feature selection (e.g. filter or wrapper), there is a potential issue that has usually been neglected in the research literature so far. This is the matter of the dataset itself.

Consider a dataset in which the important features have very nonlinear relationships with the target class. In such a case, we can expect the statistical correlation between each important feature and the target class to be very low. However, many of the standard FS methods will rate these features poorly, because of the low correlation, and may therefore not select them in the reduced dataset. To show a made-up example, consider a dataset in which there are 10 features. 8 of these features have a correlation of 0.5 with the target class, but feature 9 is always a random number, which has a correlation of 0 with the target class. We can construct feature 10, however, to be $\sqrt{\text{target\_class} - \text{feature\_9}}$. So, both features 9 and 10 may be uncorrelated with the target class and will be removed by feature selection. But, the target class can be predicted perfectly by a nonlinear combination of them, and this may not be possible with any of the other features.

This type of reasoning leads us to suspect that the right choice of feature selection method must depend in some way on statistical aspects of the dataset. For example, if the statistical correlations between the features and the target class in a dataset tend to be very low, this means that the important relationships between features in this dataset are probably not linear, and so FS methods that use statistical correlation may be a wrong choice.

Meanwhile, there are good reasons for finding a way to choose the right feature selection method in advance of running the machine learning experiments.

Assume that there are *k* indispensable features existing in an *n*-dimensional dataset, and that feature selection method 1 chooses some of these *k* features and some irrelevant features, and similarly for feature selection method 2. The only way to compare these two feature selection methods is to compare the results of later running machine learning methods on the reduced datasets. So, unnecessary time will be spent if we only need the best result.

For biologists, the discriminatory *k* features are more important than the algorithms used to select them. Usually, the experiments would be done on one or two methods.The features which belongs to the data depends on the primal dataset.

Here, we expect the relationship between the data and the correlation (the basic statistics method) could interpret how well the algorithms could be chosen in front of data. So, to get useful feature selection in general cases, and to save time, in this thesis we investigate the hypothesis that the right choice of feature selection method depends on statistical aspects of the dataset. This seems to be ignored in the literature on feature selection so far. In our review, we tend to find that papers only consider the size of the dataset, and/or the number of features. However, we investigate the idea that a statistical analysis of the data will provide additional information that will help in the right choice of FS strategy.

2.6 Brief Review of Evolutionary Algorithm

Evolutionary algorithms can be grouped into the following main groups Genetic Algorithms (GAs) (Holland J H (1975)), Genetic Programming (GP) (Koza and John R (1992)), Evolution Strategies (ESs) (Rechenberg I (1973), Schwefel H P (1981)). Each of these methods come from different independent backgrounds, but follow the same basic style of algorithm, generally known as evolutionary algorithm. They are global optimization tools which use population-based search and borrow ideas from the method of evolution in nature. They are now very popular for a very wide range of problems. Unlike many more traditional optimization methods, there is no major restriction on what type of optimization problem evolutionary algorithm can be applied to. All that is needed is a fitness function (to measure the quality of any individual solution), and a way to encode (represent) any possible solution in the search space as a vector of numbers (or in fact as any data structure). Using selection methods, and using ‘operators’ (which produce one or more offspring solutions from one or more parent solutions, for example by random mutation), an evolutionary algorithm will evolve good solutions from initial populations of randomly generated solutions. Since machine learning problems can also be seen as optimization problems (trying to find an optimised model that maximises accuracy of classification on a training set of data), evolutionary algorithms are very common in machine learning, e.g. to evolve rules, or decision trees, or other models. We use a simple evolutionary algorithm in this thesis to evolve rules, and this gives us performance measures for the FS and FB methods.

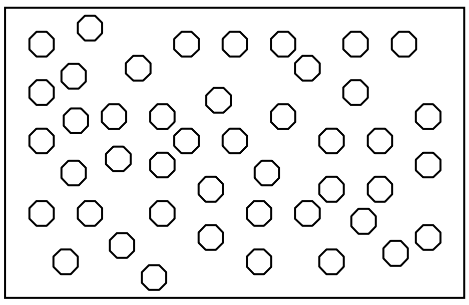
After it is decided how to encode (represent) solutions to a problem, and how to set the fitness function, an evolutionary algorithm operates the five steps:

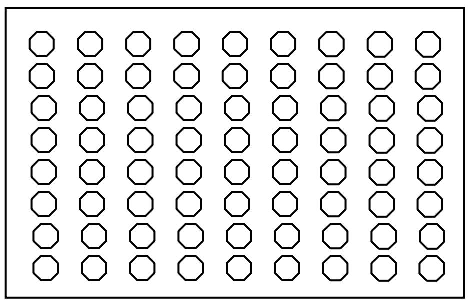
1. Create a random population of solutions.
2. Compute a fitness measure for each.
3. Select some ‘parent’ solutions from the population, and create new members by mutating and/or recombining the parents, and evaluate the fitness of the new members produced.

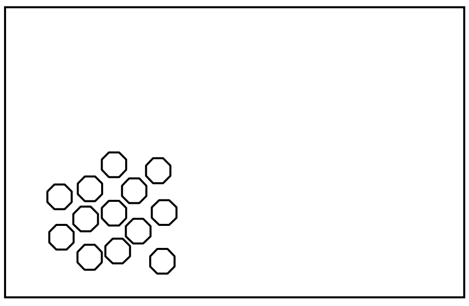
4. Update the population by including some of the new offspring and deleting some of the previous population members.
5. If a termination condition is not reached, go to Step 3.

There are several things to say, and much background research, about each of these steps. Since evolutionary algorithms are not a major part of this thesis, we do not give a long review here, but will say a little about some of the issues. For example, the first step, initialising the population, can be done in various ways as illustrated in Figure 2.7.

Usually, random initialisation is used (shown schematically in Figure 2.7 (a) for a two-dimensional real-valued search space), but we may instead use grid-based initialisation to ensure a good spread around the search space (Figure 2.7 (b)).


(a) Random Initialization


(b) Grid Initialization


(c) Knowledge-based Random Initialization

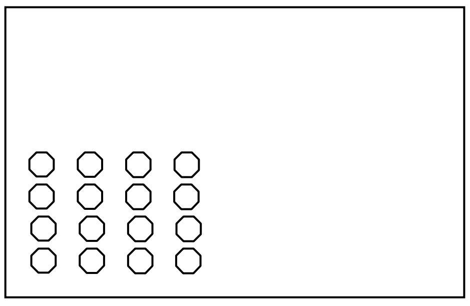

(d) Knowledge-based Grid Initialization

Figure 2.7. Four classical types of representation.

However, if we have some prior knowledge that helps generate solutions that we know will be reasonably good, we can use knowledge-based random initialisation (Figure 2.7 (c)), or the grid based version (Figure 2.7 (d)). But a problem with this is that the search may be too focused on the area around the solutions that are preferred by the prior knowledge, and this may miss even better solutions in the search space. A randomly initialised population may allow the EA to discover fundamentally different solutions in comparison with what a human would have proposed. In general, which is also true with most aspects of evolutionary algorithms, the best choice of initialization method depends on the problem one is solving. Random initialization is used in most general investigations on EA, however a good rule of thumb is to incorporate as much expert knowledge as possible in initialization, as well as operator design, but too avoid having this knowledge bias the search too much.

Crossover and mutation are the two basic operators of most evolutionary algorithms, although some types only use mutation (such as evolutionary programming). The design of specific operators depends on the encoding and also on the problem. Examples of the simple standard operators for different types of encoding (binary and real valued) are given in Table 2.1.

Crossover Operator			Mutation Operator			
One point	Before	10010 10101	Binary	Before	1001010101	
	After	10101 10010		After	1000010101	
Two point	Before	100 1010 101 001 0101 010	Real Value	Mutation Rate (0,0.2)		
	After	001 1010 010 100 0101 101		Before	0.5	0.3
				After	0.48	0.31

Table 2.1. The types of cross over operators and mutation operators.

One of the key parts of the search is the selection method, which is used to choose parents in step 3. Another (maybe different) selection method is used in step 4, to produce the next generation from the combination of the previous population and the new offspring. A selection method basically provides a way to choose a solution from the population in a random way, but biased towards choosing more fit members.

Common choices are:

- Roulette Wheel Selection
- Rank Selection
- Steady-State Selection
- Elitism
- Tournament selection

We outline tournament selection here, since that is what we use in later chapters. The following pseudocode method, for example, results in choosing a single selected individual from a population *P* by using tournament selection with a tournament size of *t size*.

Steps of tournament selection method are as following:

1. Pick *t size* individuals randomly from *P*, and place them in *T*.
2. Determine the fittest individual in *T* (breaking ties randomly), and call it *c*
3. Return *c* as the selected individual.

To select *k* individuals, for example, we simply run the above procedure *k* times.

A summary of some important general notes about evolutionary algorithms design are listed as following Table 2.2.

In conclusion, evolutionary algorithms (Spears W A, *et al.* (1993)) are general and successful global optimization tools. They are applied in a vast number of domains, and have shown good and robust performance on a broad range of real-world problems, such as automatic design, optimisation (Fogel D B, *et al.* (1999)), pattern recognition (Kudo M and Sklansky J (2000)), control and many others. They are very often used in machine learning, and they are used in this thesis as the machine learning method, in which they evolve a set of simple rules for each classification task.

Crossover Rate	Crossover rate should be high generally, about 80%-95%. (However some results show that for some problems crossover rate about 60% is the best.)
Mutation Rate	On the other side, mutation rate should be very low. Best rate seems to be about 0.5%-1%.
Population Size	It may be surprising, that very big population size usually does not improve performance of GA (in the sense of speed of finding solution). Good population size is about 20-30, however sometimes sizes 50-100 are reported as the best. Some research also shows that the best population size depends on the size of encoded string (chromosomes). It means that if you have chromosomes with 32 bits, the population should be higher than for chromosomes with 16 bits.
Selection	Basic roulette wheel selection can be used, but sometimes rank selection can be better. There are also some more sophisticated methods that change parameters of selection during the run of GA. Basically, these behave similarly like simulated annealing. Elitism should be used for sure if you do not use other method for saving the best found solution. You can also try steady state selection.
Encoding	Encoding depends on the problem and also on the size of instance of the problem.
Operator Type	Operators depend on the chosen encoding and on the problem.

Table 2. 2. General notes while using evolutionary algorithms.

2.7 Conclusion

Broadly speaking, there are two very prominent application areas in which feature selection is commonly applied. The first one is many-attribute bioinformatics datasets (e.g. gene selection from micro-array datasets (Xing E P and Karp R M (2001)), and the other is text categorization (Yang Y and Pedersen J O (1997)).

In this thesis, we focus on bioinformatics datasets, mainly proteomics data, with the task of predicting a target class. That is, the classification of these proteomics datasets have been done by bio-researchers and the problem is how to improve the accuracy of prediction of new samples. Because these datasets have usually thousands of features, the question of feature selection becomes a critical point, and is widely discussed in this thesis. As we have seen in the review, research concerned with feature selection is aimed at trying to improve the efficiency of the method, and the performance (in terms of how much it helps the machine learning classification task), and investigate new methods and test them on certain datasets. But there has been very little guidance about how to choose the right feature selection method for a given dataset, and we have found no studies at all which attempt to look at statistical aspects of the dataset to help with the choice. We claim that this is an important issue and we explore it here.

Chapter 3

Feature Selection Strategies and the Dataset Correlation Value

3.1 Introduction

Feature selection (Guyon I and Ellisseff A (2003)) is often found to be an essential pre-processing step when data mining is applied to many-attribute datasets (e.g. several hundred or thousands of attributes). In proteomics, for example, a single data sample may be a vector of several thousands of real values (representing mass/charge ratios from a spectrometer). Similarly large datasets are well-known to appear in other areas of bioinformatics (Brazma F and Vilo J (2000)), as well as a variety of other disciplines (Beynon M, *et al.* (2002)). Feature selection aims to pre-select a relatively small number of attributes, thus speeding up further processing and (hopefully) eliminating data that have minimal or no discriminatory power. Often, feature selection is done on the basis of the straightforward statistical correlation, discarding features that have the lowest correlation with the target class(es). However, when these correlation values tend to be rather low for all features (common in many datasets of importance), the basis for pre-selection of any specific set of features is unclear, and it can be imagined that straightforward feature selection may do more harm than good. In this chapter we confirm this by investigating the performance of five feature selection strategies on several datasets with varying "dataset correlation values". The dataset correlation value is a simple summary measure of the correlations between features and the target class in a dataset. We find that using a straightforward statistical correlation based feature selection method is never the best choice for poorly correlated data (datasets with low dataset correlation values). The most reliable methods among those tested, for poorly correlated datasets, are either *No Feature Selection*, or *Evolutionary Algorithm Feature Selection*.

The remainder is set out as follows. In section 3.2 we broadly describe the algorithms, datasets, and general approach taken in the reported work. Section 3.3 then describes our experiments, and reports the results. These results are discussed in section 3.4.

3.2 Algorithms and Datasets

3.2.1 The Overall Framework

In a wider context, there is much research effort looking at the use of evolutionary algorithms or similar methods to search the space of rules (and/or similar discriminatory patterns) in large-scale datasets, with emphases on bioinformatics and proteomics. In such data, particularly proteomics data, a common characteristic is that each data sample is a vector of many attributes (often several thousands). For efficient evolution of rules, it is therefore very helpful to pre-select a subset of the attributes, discarding the rest, hence reducing the size of the dataset. This reduction can lead to an immense speedup in processing time, as well as potentially much better generalization performance. Improved accuracy arises since, if we use an appropriate feature-selection strategy for this pre-selection phase, we can expect that a great deal of noise has been removed from the data and that we are concentrating on features that are more pertinent to the classification task at hand.

Feature selection is often done *a priori*, so that a machine learning method can then use a reduced dataset, but it may also be incorporated into the machine learning approach itself. As we have discussed, these are, respectively, the well-known *filter* and *wrapper* approaches (Kohavi R and John G H (1997)). In this chapter we use a simple filter approach, but we expect that the findings, with respect to choice of feature selection strategy, might be pertinent to any system that employs machine learning, especially where the datasets are poorly correlated.

The contribution of this chapter is an empirical study of how the relative effectiveness of five feature selection techniques (including the 'null' technique, in which no features are discarded) varies according to a rough measure of the degree to which the dataset is correlated. That is, for each of several datasets with various different degrees of correlation (what we mean by this is explained soon), we compare the performance of five feature selection strategies. Though feature selection comparison studies are frequently reported (e.g. by Jain A (1997), Koller D (1997) and Forman G (2003)), the issue of their performance relative to the dataset's inherent correlation seems to have been overlooked. We suspect that this is partly due to the fact that it is not easy to find suitable sets of datasets with widely varying amounts of correlation. We address this in a simple way: we choose a variety of datasets with a range of values, and we add more datasets to these by adding noise to some original datasets. By adding different amounts of noise to a dataset, on a trial and error basis, we simply produce artificial datasets with a correlation value close to any desired value.

3.2.2 Feature Selection Techniques

The null technique is no-feature-selection (NFS); when NFS is 'applied', this just means that the evolutionary algorithm rule learner (see section 3.2.4) works with the full training set.

The next technique is straightforward ranking of features based on the most commonly used standard statistical correlation measure. This is the *sample correlation coefficient*, or the *Pearson product moment correlation coefficient*. We will refer to it hereafter as the *correlation coefficient*. It is given as:

$$c_{f,p} = \frac{n\sum_{i=1}^{n} f_i p_i - \left(\sum_{i=1}^{n} f_i\right)\left(\sum_{i=1}^{n} p_i\right)}{\sqrt{\left(n\sum_{i=1}^{n} f_i{}^2 - \left(n\sum_{i=1}^{n} f_i\right)^2\right)\left(n\sum_{i=1}^{n} p_i{}^2 - \left(n\sum_{i=1}^{n} p_i\right)^2\right)}}$$

where $c_{f,p}$ is the correlation between feature *f* and the target feature *p*, *n* is the number of (*f*, *p*) value pairs in the data, while f_i is the value of feature *f* in the *i*th such pair.

The third technique we test is a variant of the well-known Relief-F method by Kononenko I (1994) , which was designed to cope well with noisy data (hence we have a prior expectation that Relief-F may be better on poorly correlated datasets). The procedure for obtaining the Relief-F value for each field is as follows:

1. Set all feature weights w_i to 0.

2. Calculate the target class probabilities – for two-class data, we denote these respectively as *p* and *q*, respectively the proportion of positive and negative records in the dataset.

3. Repeat the following for *m* iterations (we use *m* = 0.1 times the number of fields).
   a. Choose a random data record *R*.
   b. Find the sets *Rp* and *Rn*, respectively the 10 records closest (in Euclidean) to *R* which are in the positive class, and the 10 records closest to *R* in the negative class.
   c. For each field *i* set: $w_i = w_i - X$, where *X* is:

$$(1/10m)\left(\sum_{j=1}^{10} d(i,R,Rp_j) + \sum_{j=1}^{10} d(i,R,Rn_j)\right)$$

Where Rp_j denotes the *j*th record in the set Rp , and $d(i,R,Rp_j)$ indicates the absolute difference between the value of attribute *i* in record *R*, and in record Rp_j, as a proportion of the range of values in the dataset for attribute *i*.

The resulting set of feature weights become the Relief-F values for each feature, with higher values indicating more usefulness in helping discriminate between values of the target class. Clearly, Relief-F is designed to be more sensitive than the correlation coefficient to feature interactions. Relief-F in particular is especially appropriate for multi-class target attributes, which we are interested in for subsequent work, however these aspects are not dealt with in the preceding pseudocode, which specifies how we use it in the 2-class cases studied in this paper.

The fourth technique we test is to use an Evolutionary Algorithm (EA) to do the feature selection step. Using the EA as a separate feature selection step is a relatively little explored idea, however it has been found successful in recent work. The idea is to apply a short EA run, using precisely the same rule-learning EA detailed later, and collect features from those that appear in rules in the final population of that EA. The surviving attributes are, almost by definition, ones that are likely to have useful discriminatory value, either alone or in combination with other attributes. When EA is the feature selection technique, we ensure that the computational cost is taken into account, in the comparative studies, by appropriately reducing the number of generations then allowed in the next stage.

Our final feature selection technique is simply a combination of the correlation coefficient and Relief-F, which we denote CRFS. Having obtained correlation values for each feature for each of these methods, we simply sum the ranks of those values, and this gives us the CRFS value.

3.2.3 Datasets and Data Correlation

This work was motivated by studies on two proteomics datasets, respectively the ovarian cancer dataset from Petricoin E, *et al.* (2002), which we denote OV, and the pancreatic cancer dataset from Hingorani S, *et al.* (2003), which we denote PA. OV is relatively highly correlated – that is, when we look at the simple statistical correlation of a feature with the target feature, the values tend to be quite high. In contrast, PA has rather low correlation values. These issues are reflected in the performance that tends to be achieved on these datasets in machine learning studies – typically 95—100% accuracy on test data for OV, but 60—65% for PA.

a) Ovarian Cancer:

This research is quite significant for women, especially who have a poor risk of ovarian cancer due to their family history. How to find a proteomics pattern which can distinguish ovarian cancer patients from normal patients, becomes the purpose of research on this dataset. This dataset (each value representing mass/ charge ratios from a spectrometer) consists of 91 control samples (Normal) and 162 ovarian cancer samples, with the task being to train a classifier (in our case, learn a set of rules) to correctly predict the class of an unseen sample. The dataset is separated into 128 training samples and 125 test samples.

b) Pancreatic Cancer:

This particular dataset comes from mouse samples, developed as part of research to firstly generate a mouse model of PanIN (Pancreatic intraepithelial neoplasias). This led to a reliable means of detecting PanINs in the serum proteome of mutant animals. These results are pertinent to an accurate prediction model of the earliest stages of human neoplasias. The PA data has 181 samples divided randomly into train and test sets by us, and there are 6776 genes (features) in each sample.

When we consider the individual statistical correlation values (correlation with the target feature) for features in these two datasets, the difference is clear. Note that we consider the absolute value, so that high values (near 1) mean a strong correlation or anti-correlation with the target feature, and low values (near 0) mean very poor correlation. In the OV dataset, the highest individual feature correlation coefficient (which we call the Dataset Correlation Value (DCV)) is 0.896, while in the PA dataset it is 0.185. Various suggestions follow from such a clear distinction between these datasets. In general we should not expect that the ideal analysis method for the

OV dataset will correspond to the ideal analysis method for PA, while, of course, the potential accuracy of predictive models is possibly limited at the outset by these correlation values (although, it is entirely possible that strong predictive models are possible for the PA dataset which exploit underlying patterns that are obscured or ignored by simple pairwise correlation).

Our specific interest here is feature selection, and how the choice of feature selection method might be guided by a simple measure of the inherent correlations in the data. As mentioned, we characterise a dataset's inherent correlation in terms of this highest individual (absolute) correlation coefficient of its non-target features with the target feature, and we call this the Dataset Correlation Value (DCV). Thus, the OV dataset has a DCV of 0.896, and PA has a DCV of 0.185. This characterization is sufficient for the purposes in this chapter, however it is an open question whether the median correlation coefficient or some other averaging measure will be more generally useful. We investigate that question in a later chapter. In the cases studied here, the maximal value tended to be a good guide, rather than an outlier.

In order to investigate the relationship between feature selection method and dataset correlation value, a number of other many-attribute datasets were obtained in addition to OV and PA. To keep things straightforward, we looked for many-attribute datasets that had only real-valued features and a natural two-class classification task. However, the range of DCVs among these datasets was still quite small. After this brief investigation, two datasets were added to this study, respectively the Ionosphere and Optical Digit datasets from the UCI repository (Asuncion A and Newman D J (2008)). We decided that a fast way to obtain test datasets that had a wide range of DCVs, spanning from very low to very high, was to artificially add noise to an existing dataset that itself had a high DCV. The best candidate for that was the OV dataset. We therefore generated several variants of the OV dataset by adding different amounts of noise to the attributes. The result was an additional 11 datasets that we called rOV1, rOV2, …, rOV11. These are shown in Table 3.1, along with the four original datasets, listed in ascending order of DCV.

DCV	Dataset
0.099	rOV1, 1000 fields
0.185	Pancreatic, 8,642 fields
0.335	rOV2, 1000 fields
0.349	rOV3, 1000 fields
0.378	Opt digit, 64 features
0.399	rOV4, 1000 fields
0.449	rOV5, 1000 fields
0.496	rOV6, 1000 fields
0.51	Ionosphere, 32 features
0.539	rOV7, 1000 fields
0.598	rOV8, 1000 fields
0.618	rOV9, 1000 fields
0.699	rOV10, 1000 fields
0.784	rOV11, 1000 fields
0.896	OV, 15,143 fields

Table 3.1. Datasets used in the experiment of chapter 3.

In order to generate the rOV datasets, we first chose the top 1000 features from OV according to correlation coefficient with the target class. Then, we produced a new randomised OV (rOV) dataset by adding a small random value to each field of each feature, and calculated the new dataset's DCV. This entire process was repeated for increasing values of the random value's range parameter, until datasets were acquired with correlation values close to 0.1, 0.3, 0.6, 0.7 and 0.8 (existing datasets were available with values already close to 0.2, 0.4, 0.5 and 0.9). In this way we got the eleven additional datasets used later in this chapter and later in this thesis.

3.2.4 The Evolutionary Algorithm Rule Learner

The 'quality' of a feature selection method in respect of a dataset can be estimated by evaluating the performance of a model learned, by any given machine learning method, from that dataset, considering only the selected features. In our work the machine learning method that we choose is an evolutionary algorithm that learns simple rules that relate the relative values of different attributes. Here we briefly describe our evolutionary algorithm (Fogel L J, Owens A J and Walsh M J (1966)) for learning rules in these data. It is deliberately a simple and straightforward approach; the research was not concerned with finding the best performing design for an evolutionary algorithm rule learner. The requirements were only to define a suitable machine learning method that could be used for experiments about comparing feature selection methods.

We evolve "IF THEN" rules with a fixed number of antecedents (in all cases in this chapter, five antecedents). The meaning of a rule with antecedents "A B C" is "If A and B and C are all true, then this data sample is a Positive case, OTHERWISE it is a Negative case". For example, when the target field is "cancer" or "normal" (as in the OV dataset), "cancer" is the positive case. By including OTHERWISE in the semantics of a rule, we avoid issues of coverage – that is, the rule specifies a class for any data sample, not just those that meet the antecedent conditions; this simplifies the evaluation function, which is merely the rule's accuracy on the training set.

An individual antecedent is of the form:

$$[f_X, f_Y, C]$$

where f_X and f_Y denote fields in the data, and *C* is a comparator, either ">" or "<". Hence, the following example rule (encoded as a list of antecedents):

[[2184, 781, >], [30, 2844, <], [101, 22, >]]

encodes the rule: "If the value of field 2184 is larger than the value of field 781, and the value of field 30 is smaller than the value of field 2844, and the value of field 101 is larger than the value of field 22, then this sample is a positive case, otherwise it is a negative case".

When a rule is mutated as part of the evolutionary algorithm, a gene is chosen at random, and then altered at random. E.g. we may choose the comparator in one of the antecedents, and change it, or we may choose (usually) a field identifier and change it. In the current work, we use a steady state evolutionary algorithm with a population of 100, and the mutation operator incorporates a brief local search. Specifically, following the generation and evaluation of a random initial population, the following is done in each iteration. A single parent is chosen by binary tournament selection. We then generate ten mutants of this parent; the fittest of these mutants then enters the population, overwriting the current least fit in the population. This continues for a given number of cycles. In the experiments reported, the number of iterations is 1,500. When EA is used as the feature selection scheme, the FS EA runs for 500 iterations.

Basic information about the EA is summarised in following table:

Population	100
Antecedents Per Rule	5
Maximum Iteration	1500
Tournament Size	5
Operator	Mutation
Offspring	10

Table 3. 2 Summary of EA parameters.

3.3 Experiments

3.3.1 Comparison of Five Feature Selection Methods

Feature selection methods were applied in the following way. In the case of no-feature-selection (NFS), we naturally did no feature selection! For each of the correlation coefficient (CFS), Relief-F (RFS) and combined (CRFS) methods (see later), a subset of *N* features was obtained for each dataset. For the EA (EAFS) method, a 1,000 iteration run of the EA was performed, and the *N* most-referenced features in the final population were collected, and these became the selected features.

For each dataset *D* and each feature selection technique *F*, the following experimental design was then used in every experiment reported below. In all cases except those of

Ionosphere and Optical Digits data, *N* was 100; in the former two cases, *N* was 10 and 30 respectively.

D was divided into a training set and a test set, by a random 50/50 split. Feature selection method *F* was then applied to the training set, resulting in a ranking of the features. The training set was then reduced by extracting only the selected *N* features for each sample. The following was then repeated 5 times with independent random seeds: The evolutionary algorithm was run on the reduced dataset, returning a single rule that performed best on the training set. The performance of that rule was then measured on the test set, and this was recorded as the performance value for this trial.

CFS	**RFS**	**CRFS**	**EAFS**	**NFS**	**DCV**
2	5	5	1	3	0.099
5	4	*3*	2	1	0.185
4	1	5	*2*	3	0.335
2	5	4	3	*1*	0.349
5	4	1	3	3	0.378
4	5	1	3	2	0.399
3	5	4	1	2	0.449
3	5	2	1	4	0.496
1	5	4	3	2	0.51
1	4	5	3	2	0.539
1	5	4	2	3	0.598
1	4	5	3	2	0.618
3	5	4	2	1	0.699
1	5	4	3	2	0.784
1	4	5	3	2	0896

Table 3.3. The rank of performance (1 = best, 5 = worst) per dataset for each feature selection on each of the datasets, represented by correlation value.

Following 10 trials for a specific dataset/feature-selection strategy pairing, the mean result is recorded as the summary result for that experiment. Table 3.4 records these summary results, in which all the values are rounded to one decimal place. These are displayed graphically in Figure 3.2. Meanwhile, Table 3.3 provides an alternative view in which we list the rank (from 1 (best) to 5 (worst)) in terms of performance for each correlation value, and these are shown graphically in Figure 3.1.

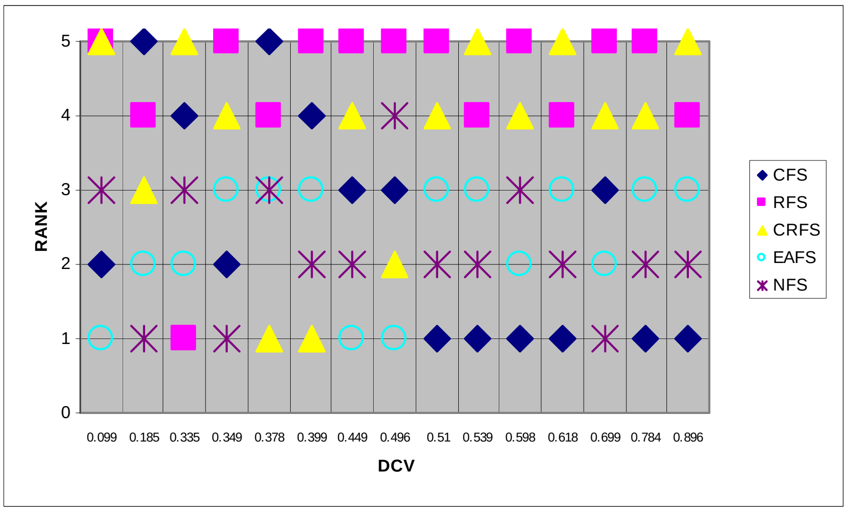


Figure 3.1. A scatter plot of the rank (from Table 3.3) and DCV for each FS method.

As shown in Table 3.4, the accuracy varies much as the DCV is increased. Some of the better accuracies reach 90%, which is not attained in the cases of the four lowest DCV values by any of the methods which reduce the feature set. When the DCV is high, the FS methods can be expected to retain features that correlate very well with the target, and the EA may find it easy to build suitable rules that exploit these features. However, for low DCV datasets (e.g. 0.19), it is reasonable to expect that FS methods are very challenged to find a suitable reduced set of features for later machine learning. We can see this reflected in Figure 3.1. For example, the blue diamond appears consistently towards the lower right of the figure (good rank and high DCV), but tends to perform poorly for low DCV.

When we look at raw results, we note (though this is not the major point of the current work) that the performance of the basic EA rule evolution strategy used here is not very different from reported performance for the established (i.e. non-rOV) datasets here. These are rows 2 (PA), 5 (opt digit), 9 (Ionosphere) and 15 (OV).

CFS Test	RFS Test	CRFS Test	EAFS Test	NFS Test	DCV
79.4	69.1	69.1	80.2	78.4	0.099
46.9	46.0	50.2	50.5	56.9	0.185
63.2	64.2	62.4	64.0	63.4	0.335
89.8	83.0	87.5	89.1	91.7	0.349
85.3	88.8	90.0	89.8	89.8	0.378
86.2	84.2	90.7	89.8	90.2	0.399
86.2	82.7	85.6	91.0	88.6	0.449
90.2	87.5	90.7	91.0	89.0	0.496
91.5	64.6	78.1	85.3	91.3	0.51
77.6	65.3	65.1	74.6	75.5	0.539
93.2	86.4	86.6	92.3	90.2	0.598
85.0	65.8	65.0	74.9	77.1	0.618
87.0	75.2	81.0	88.0	90.4	0.699
84.2	68.0	68.6	79.7	82.6	0.784
98.9	83.7	81.0	89.0	92.2	0.896

Table 3.4. Summary of test results for each feature selection on all the datasets, represented by DCV.

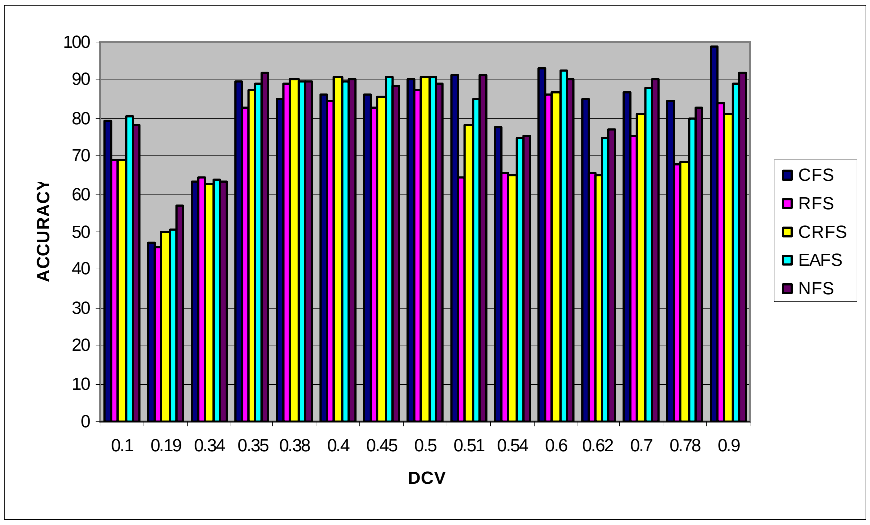


Figure 3.2. A graphic presentation of the ranks in Table 3.4. The real accuracies of five algorithms are compared for each DCV in the figure.

However a main issue of interest is the deterioration in performance of CFS as the DCV reduces. Though it is often used as a feature selection method, especially in bioinformatics research, CFS clearly seems to be a poor strategy when the dataset's correlation value (as defined here) is below 0.5. For such poorly correlated datasets, the best strategy (among those tested) seems to be to use either EAFS or use no feature selection at all. Meanwhile we are surprised at the quite poor performance of Relief-F overall. This can partly be explained by the method we used to reduce correlation in several of the datasets – Relief-F can be expected to pick up on some feature interactions that are not picked up by standard correlation based feature selection, however by adding randomness we did not introduce any such interactions. But, Relief-F's performance was relatively poor in all cases except one. This perhaps underlies the poor performance also of CRFS, although there does seem to be evidence that CRFS, or similar combined strategies, have a niche of performance with regard to datasets with a small range of low–medium correlation values.

From one viewpoint, considering the rankings in Table 3.3, EAFS appears the most successful of these techniques, since it was never worse than third ranked in any case (all other methods, though NFS has similar overall performance).

3.3.2 A Note on Preliminary Experiments with the EA Rule Learner

In this subsection we briefly mention the experiments that were done to decide certain of the main parameters for the EA rule learner. Although not central to this thesis, this may be of interest to readers. The main experiments done were to determine the fixed number of antecedents for the rules. In general, the number of antecedents does not have to be fixed; however, that would mean a more flexible but more complicated EA, with several more parameters and design issues to adjust (for example, different genetic operators that increase or decrease the number of antecedents, and probabilities for those operators). By choosing a simple EA with fixed number of antecedents, we avoid complications in interpreting the results, so that differences in performance are more likely to be a result of the different feature selection methods, instead of the result of complex interactions between the problem and the EA design.

As we mentioned last section, an individual in the population consists of five antecedents, for example:

$$\{[f_{X1}, f_{Y1}, C_1][f_{X2}, f_{Y2}, C_2][f_{X3}, f_{Y3}, C_3][f_{X4}, f_{Y4}, C_4][f_{X5}, f_{Y5}, C_5] \Rightarrow Cancer\}$$

This rule infers the positive case "cancer" (e.g., in OV) if the five antecedents, each a Boolean expression, are all true. Obviously, if rules were only allowed to have one antecedent, we could only get accurate rules if some features existed which have a very strong correlation with the target class.

Rules with several antecedents are, however, capable of representing interactions between features that lead to a better chance of classifying accurately in difficult datasets (e.g. CITE, CITE, CITE). The question naturally comes up: how many antecedents shall we have in our rules? Experiments were done as follows to guide this choice.

DCV	**CFS Test 1**	**CFS Test 5**	**CFS Test 10**
0.099	80.8	79.4	**82.4**
0.185	**55.5**	46.9	51.7
0.335	56.8	63.2	**64.6**
0.349	88.8	**89.8**	88.2
0.399	85.4	**86.2**	85.7
0.449	**88.0**	86.2	85.9
0.496	**93.6**	90.2	87.0
0.539	76.2	77.6	**79.0**
0.598	92.6	**93.2**	93.1
0.618	68.8	**85.0**	75.4
0.699	82.2	87.0	**92.0**
0.784	83.2	**84.2**	82.2
0.896	**100**	98.9	95.4
Average	80.9	**82.1**	81.7

Table 3.5 The performance of CFS with one, five, ten antecedents on all datasets. The best mean accuracy in a row is highlighted in bold.

To observe the influence of the numbers of antecedents, we first tested the performance of CFS with three different numbers of antecedents used in the evolutionary algorithm rule learner. In Table 3.5 we record the results of these experiments for one antecedent, five antecedents, and ten antecedents. The results are the mean accuracies on the test set of five independent runs each on a selection of the datasets we have discussed, identified in the table by their DCVs.

As exhibited in Table 3.5, the average performance of CFS with 5 antecedents is better than these of other two conditions. Also, the number of times a condition achieved the best over the three conditions is highest for five antecedents. The differences are not very large, however another consideration is computational cost. Checking the accuracy of a rule increases in time complexity with the number of antecedents. Also, considering that using one-antecedent rules would clearly be limited in potential (and anyway the space of one-antecedent rules is better explored by brute force search), the suggestion from these experiments is that five-antecedent rules may be the best choice of the three.

DCV	**EAFS Test 1**	**EAFS Test 5**	**EAFS Test 10**	**NFS Test 1**	**NFS Test 5**	**NFS Test 10**
0.099	79.4	**80.2**	75.4	**85.9**	78.4	70.7
0.185	**58.6**	50.5	48.6	54.5	**56.9**	54.0
0.335	54.4	64.0	**64.8**	50.9	63.4	**64.8**
0.349	89.6	89.1	**91.0**	89.1	**91.7**	87.6
0.399	88.8	**89.8**	89.0	87.6	**90.2**	82.9
0.449	89.1	**91.0**	89.6	87.4	**88.6**	87.8
0.496	82.4	**91.0**	87.8	86.9	**89.0**	87.2
0.539	74.2	**74.6**	70.9	69.8	**75.5**	66.1
0.598	90.4	**92.3**	90.5	89.4	**90.2**	88.8
0.618	68.5	**74.9**	74.4	72.3	**77.1**	69.3
0.699	80.0	88.0	**89.6**	85.8	**90.4**	87.0
0.784	79.2	**79.7**	74.1	82.2	**82.6**	82.2
0.896	**95.1**	89.0	87.6	**93.9**	92.2	90.4
Average	79.2	**81.0**	79.4	79.6	**82.0**	78.3

Table 3.6. The performance of EAFS and NFS with one, five, and ten antecedents rules on test data. Each row has two bold numbers, showing the best result for each of EAFS and NFS.

We also tested these three conditions with EAFS and NFS, to see if this conclusion can be validated. The results are in Table 3.6, where "Test*N*" means the mean accuracy on the test set when the evolutionary algorithm rule learner used *N*-antecedent rules.

It is very clear from Table 3.6 that the performance of five antecedents in a rule is much better than these from one antecedent and ten antecedents. We therefore chose five-antecedents for all later experiments.

3.3.3 A Note on Preliminary Experiments for Choosing the Number of Features

Another important parameter to fix is the number of features that will be selected by the feature selection methods. If only a very small number of features is chosen (e.g. 5 or 10 features from a dataset with many thousands of features), the results might be over-influenced by random chance effects. If very many features are chosen, we possibly lose the ability to discriminate well between the algorithms. We therefore did some preliminary investigation of the number of features as follows. We chose seven different cases of number of features to investigate on the OV dataset, and looked at the performance of CFS (accuracy on the test set averaged over five runs). Table 3.7 shows the results, also showing the mean accuracy on the training set. The numbers “36”, “65” and “783” look like strange choices of values to set for the number of features. The explanation for these comes from other preliminary experiments, where we were using EAFS as the feature selection method and different sizes if the population and different numbers of rule antecedents. In the EAFS case when used for feature selection, a short EA run is done, and all the features that appear in the rules in the population at the end of this run are the chosen features. These numbers were therefore chosen for CFS experiments that could directly compare with corresponding CFS experiments.

As shown below, we clearly see the results from correlation features are good until around 300 features or more are selected. Clearly the most successful number of features, for this limited test, is 100.

Features	36	65	100	200	300	500	783
Performance train	99.3	99.6	**100**	99.8	99.8	99.6	98.9
Features	36	65	100	200	300	500	783
Performance test	97.28	97.76	**98.88**	97.44	95.52	94.88	92.48

Table 3.7. The performance of correlation feature selection on ovarian data. The accuracies of our features are displayed in performance train and test.

There are several factors that will influence this result. E.g. when a large number of features is chosen, the EA run may need more time (i.e. more fitness evaluations) to approach its optimal performance. But, given a desire to limit the computational time needs of future experiments, we decided from this test to fix the number of selected features at 100. Most experiments that were discussed in this chapter and elsewhere in this thesis therefore use 100. However on some datasets the number of attributes is small and these are treated differently, as we will say in the text.

3.4 Conclusion

Feature selection is a very important part of many current commercial and scientific data mining tasks. When searching for biomarkers among many thousands of possibilities, for example, successful work will lead either to new accurate diagnostic tests, or in many cases novel and actionable insight relating to the specific sets of features found to be most important. But, given the enormous number of attributes in many cases, computational limitations require researchers to pre-select a small set of features, and this is normally done by using straightforward statistical measures of correlation (such as CFS). However, in such datasets the interactions between the features themselves and the target class(es) are rarely simple linear interactions, and we believe that many standard techniques in use will often discard important features, leading to lost opportunities in terms of predictive accuracy and scientific insights.

In the work reported here, we have investigated the performance of a small range of feature selection strategies, and find that their relative performance is related to the degree of correlation between features and the target class in the underlying data. CFS appears to work very well when the DCV is high, but it is also clear that CFS is never the best choice, and often a poor choice, for datasets where the DCV is low. In particular, we can claim here some tentative steps towards choosing a feature selection technique based on the simple prior calculation of DCV. However, much work remains to investigate what strategies are most useful for poorly correlated data, especially given that many of the poorly correlated datasets in this chapter were generated by random perturbation.

The limitations of the work described in the chapter include the fact that we only use a simple summary measure to characterise a dataset (DCV), the use of randomization in creating datasets with specific correlation values (which means they don’t necessarily reflect deeper structure in real datasets that may share the same value), and the fact that just a limited number of FS strategies were tested. However we can claim to have confirmed that using the standard statistical correlation coefficient as an FS strategy is potentially harmful on poorly correlated datasets, and that EA-based feature selection strategies are worth further exploration. Variants of Relief-F and combinations of that with standard correlation also seem to have some value for certain levels of DCV.

Chapter 4

Investigating Feature Bias Strategies

4.1 Introduction

Previous experiments indicate that the major problem in the original proteomics data (OV and PA) is that too many genes are included, and some of them are irrelevant to the analysis. An FS strategy is often found to be an essential step to erase some useless features, thus reducing the evolution processing time and improving the performance. However in some poorly correlated datasets, we also see that well-known feature selection methods are not the best approaches. Sometimes, a no-feature-selection (NFS) strategy gives better performance. This raises the question, why is the performance of an FS strategy worse than NFS in these cases?

The idea of FS methods is to find “good” features, usually identified by correlation-based measures of features with the target attribute, which have more discriminative power than others. Often only a small percentage of features are finally used in machine learning process, saving computation time and improving accuracy. However, alternative FS strategies can vary much in the features that they end up selecting. For example, in our tests on proteomics data we find that the overlap between features selected by different FS methods tends to be small. In particular, a feature that is ranked very highly (perhaps at the top) by one algorithm will often rank in a mediocre position according to another FS algorithm. The only clear standard to estimate the relevance of those features to the learning task in question is the ultimate performance after applying machine learning to find a predictive model. In such tests (including some reported in this paper), it is commonly found that features ranked poorly by some (or even all) FS methods, still seem very pertinent to the classification task. That is, when a “no feature selection” strategy is applied in some cases, a better predictive model is found, which makes use of features that were considered irrelevant by FS methods.

However, NFS is troublesome as a general strategy because it is certainly not always the best approach. First, it loses the computational speedup that is available if we do FS. Second (and particularly on highly correlated datasets) the choices made by FS methods tend to be appropriate, so NFS slows the process down with little effect. Instead, we describe here the alternative *Feature Bias* (FB) strategy. Like NFS, FB allows the machine learning algorithm to use any feature at all – so, all of the pertinent features are available; unlike NFS, FB does not throw away completely the guidance that can be obtained from an FS method.

A Feature Bias strategy is therefore a different approach to “feature management” (FM) which basically inherits and extends the idea of feature selection. The concept of FB is to add some finer control elements to the interface between the dataset and the machine learning method. In feature selection, the control is essentially binary – for any feature, the FS stage either says “use this feature”, or “do not use this feature”. In FB, the control element is to provide a bias value. So, after applying FB to find bias values, the machine learning method might still use all of the features in the dataset, but it will be guided by the bias values towards more preferably using some features rather than others.

The idea of investigating FB strategies is chosen because we expect it to improve upon basic FS strategies, especially for datasets with low DCV. In these datasets, we can easily expect CFS (for example) to lose some important features, but in a Feature Bias strategy, all features will be available. At the same time, we were very interested to see if the relationship between DCV and FS performance, seen in chapter 3, was still there if we use FB strategies.

The main contribution of this chapter is an empirical comparison study of several FS and FB techniques; a secondary contribution is to note how the relative performance of these methods varies according to the DCV. The latter contribution extends to FB strategies the findings in Luo S and Corne D W (2008) (and chapter 3), which showed that the DCV can provide clear guidance towards the likely performance of FS methods relative to NFS.

As we mentioned, the datasets we are interested in have many attributes (thousands), and the search space for machine learning methods is very large. If no feature selection is done, the time for computation may be prohibitively expensive. The same is true for FB methods, in which (by default, in this work), all features are still retained in the dataset. But we expect FB to still have an advantage over NFS, because although the background processing time may still be large because of the many features, the guidance provided by FB should mean that the machine learning method arrives at good solutions more quickly than NFS. We take notice of this issue and ensure that our experiments compare FS and FB on the basis of a similar level of computation cost.

In the next section, we will briefly overview FS strategies and present three FB methods to compare against the original FS algorithms by applying them to several datasets: The FB strategies are: Correlation Feature Bias (CFB), Relief F Feature Bias (RFB), Evolutionary algorithm Feature Bias (EAFB) and two combined methods: CFS / CFB, and RFS / RFB. Section 4.3 will give an overview of datasets used in this chaper. Experiments and analysis will be described in section 4.4. Conclusions and future work are discussed in section 4.5.

4.2 Feature Bias Methods

In feature selection (FS), we divide the features into two sets, the selected features, and the eliminated features. The eliminated features are removed, and play no part in the machine learning. In a pure feature bias (FB) strategy, there are no ‘eliminated features’; instead, the first set is called ‘preferred’, and the second is called ‘non-preferred’. All features may be used in the machine learning. However, in FB, some features will be preferred over others.

The Feature Bias strategies work through two parameters to control the selection of features. The first parameter, *x*, indicates what proportion of the features to use; the second parameter, *y*, indicates (when the machine learning method is running, and needs to choose a feature) the probability of choosing a feature from the selected set. For example, if we use an FS algorithm to obtain 100 features out of 10,000 features, this is the same as using Feature Bias like this: the first parameter $x = 100 / 10{,}000 = 0.01$ (select 1% of the features via FS), and the second parameter $y = 1$ (with probability 1, choose one of the selected features). Alternatively, FB with parameters (0.02, 0.8) would operate as follows: using FS, 2% of the features are pre-selected as preferred features; when the machine learning method is running, and a feature needs to be chosen (for example, to add a new feature to a rule), there is a probability 0.8 that the chosen feature will be a pre-selected one, and there is a probability 0.2 that it will be one of the non-pre-selected features.

4.2.1 Basic Feature Bias Methods

Three basic feature bias strategies are used to compare with the experiments from feature selection methods: CFB is a similar strategy to CFS, based on straightforward statistical correlation; RFB originates from the Relief F method, and EAFB is based on EAFS. With the FB strategies, the first step is the same as the corresponding feature selection method: the features which are likely to have discriminatory power are chosen by each algorithm. Such as in the Correlation-based strategy or the Relief F method, the features with higher correlation or relief values indicates more discriminative power; the first step in an FB strategy is simply to calculate, as before these basic values for each feature, just as is done in the FS strategies (see Figure 4.1).

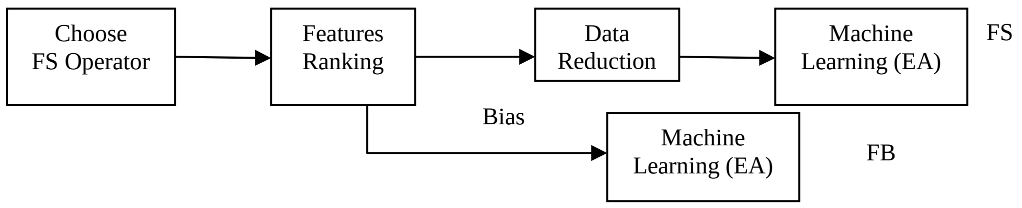


Figure 4.1. The difference between FS strategies and FB strategies.

The second part is, just like with NFS, we apply the evolutionary algorithm to learn rules (this could of course be any other machine learning algorithm) on the full (all features included) training set. In the mutation step of the EA (see Figure 4.2), we use an equation (1) to guide the selection of new features to include in a rule:

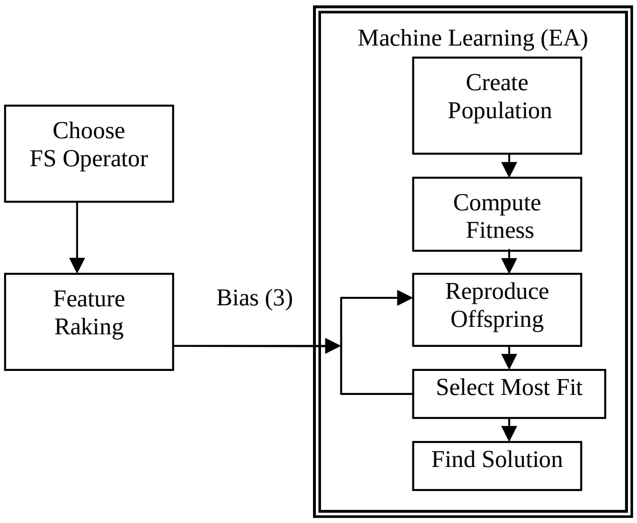


Figure 4.2. The procedure of FB strategy being embedded in the evolutionary algorithm.

$$offspring = f(x, y) \qquad \{x \in [0,1], y \in [0,1]\} \tag{1}$$

Where offspring is the identification number of a field in the dataset, and *x* and *y* are parameters. The first parameter *x* is the proportion of the features which are to be in the preferred set of features. If $x = 0.1$, and there are 1000 features in the data, that means 100 features will be in the preferred set. In CFB, these would be the top 100 features according to correlation value; in RFS, these would be the top 100 features according to the Relief F value, and in EAFB, these would be the 100 features selected on the basis of a previous EA run. The second parameter *y* is the probability of choosing a feature from this preferred set. E.g. If $y = 0.6$, that means, there is 60% chance to select a feature from this set, and 40% chance to select a feature from outside the preferred set (e.g. any other features).

4.2.2 The Combined Methods: CFS / CFB, RFS / RFB

These two strategies can be seen as combinations of feature selection and feature bias methods. In CFS/CFB, before we do the FB strategy, we still use FS to reduce the dataset, but we reduce it by a relatively small amount. Taking the ovarian data as an example, we cut off over 90% irrelevant features in the FS step (saving 1,000 features from 15,154). And an FB strategy will be applied on this 1,000 features dataset. In contrast to the original FS strategies which discard more than 99% of the features, many more features are available at the machine learning stage.

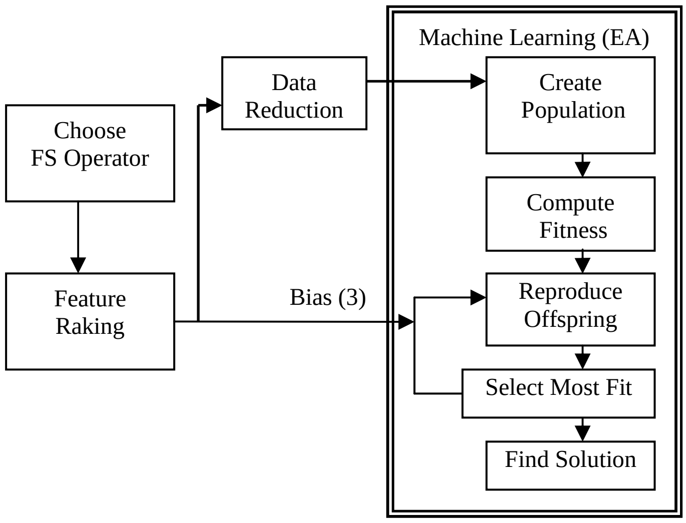


Figure 4.3. The procedure of Combined FS and FB (CFS/CFB) strategy.

With the CFS/CFB method, we do the CFS first, then apply the correlation feature bias strategy. RFS/RFB uses the Relief F method for initially selecting features, and then operates as RFB with the reduced feature set. These strategies attempt to use the advantages of both feature selection and feature bias; that is, in comparison to FS, they leave more features available to the machine learning stage and hence are less likely to miss pertinent features; in comparison to pure FB, however, there is saving in computation time because some features are removed.

4.3 Datasets and Data Correlation Values

4.3.1 Datasets

Three more proteomics datasets and nine Optical digit datasets are now added for the comparison between FS/FB strategies. Ultimately, in our experiments, we obtain 29 datasets which can be divided into three groups: 1) The proteomics datasets; 2) Reduced ovarian datasets; 3) Some small numeric datasets. The reasons for choosing these datasets are their predominance in the literature and the prevalence of numeric features. As we mentioned, the study is motivated by large-scale datasets. Because of that, we put more attention on the experiments from proteomics datasets and reduced ovarian datasets. However, the results from the other datasets could be seen as complementary for our conclusions. The investigation of small datasets can be seen as another interesting topic for the comparison of FS/FB strategies.

Proteomics Data

The proteomics datasets include the ovarian dataset (denoted OV), pancreatic dataset (denoted PA) ,the Leukemia dataset (denoted AML/ALL) ,the lung cancer dataset (denoted LUNG), and the Central Nervous System Embryonic Tumors, (denoted CNS). The dataset information is summarised in Table 4.1:

Dataset	DCV	Samples	Genes	Classes(number)	Train/Test	Dataset
AML/ALL	0.828	72	7129	All(47)/Aml(25)	38/34	AML/ALL
OV	0.896	253	15154	Normal (91)/Cancer(162)	128/125	OV
PA	0.185	181	8642	Pan In(97)/Control(84)	80/101	PA
LUNG	0.882	181	12533	ADCA(150)/MPM(31)	32/149	LUNG
CNS	0.602	60	7129	Survivor(21) /Failure(39)	30/30	CNS

Table 4.1. The proteomics datasets.

These datasets are collected from online sources. The DCV value mentioned in the table represents the highest correlation value in the datasets; that is, when we measure the basic statistical correlation between any feature and the target class, this is the highest value for any feature. Classes represent the original (target) class name from dataset and shows how many examples are in each class. Train/Test is how we divided the samples. Some divisions are made by us where no guidance was available from papers or other sources, and in other cases the original source already provided a Train/Test split.

Proteomics datasets and gene expression datasets, which both have the capability to lead to reliable discrimination between different classes of patients, have attracted tremendous interests among bio-informaticians. Proteomics is the systematic large-scale analysis of protein expression under normal and perturbed states, and generally involves the separation, identification and characterization of all of the proteins in a cell or tissue sample. As the high-throughput (HT) data acquisition technologies, such as DNA microarray, protein microarray, mass spectrometry, tissue microarray, 2D gel or fluorescent microscopy, have being increasingly used in the proteomics areas, advanced analysis in these HT data is necessary to be fostered and revisited to meet the important needs of the wider biomedical community. These technologies allow scientists to monitor the whole genome or a single gene, viewing the interactions among thousands of genes simultaneously, thus with hopes of improving the accuracy and speed of cancer classification. However some common characteristics of both proteomics data and gene expression data, which explain our interest in this data for this thesis, are:

1. Each data sample is a vector of many attributes (often several thousands).
2. Given a particular machine learning task, most of the attributes are probably irrelevant to the analysis.

The following is a description of all datasets in details.

The ALL/AML Leukemia Dataset

This dataset (Golub T R, *et al.* (1999)) is a collection of gene expression profiles of 72 bone marrow samples, over 7129 probes from 6817 human genes. The data, based on gene expression monitoring by DNA microarrays, was divided into a training set of 38 samples, and a blind testing set of 34 samples by the biologists. The task is to classify two types of leukemia: Acute Myeloid leukemia and Acute Lymphoblastic leukemia. Both obtained from acute leukemia patients at the time of diagnosis. The raw gene expression data can be found at *http://www.stjudereasearch.org/data/ALL1*.

Ovarian Cancer and **Pancreatic Cancer** and **Reduced Ovarian Datasets**

(See Chapter 3).

Lung Cancer Dataset:

Lung cancer (Gordon G J, *et al.* (2002)) is the most common cause of cancer death. The traditional methods for the distinction between malignant pleural mesothelioma (MPM) and adenocarcinoma (ADCA) of the lung could be cumbersome. The new technique, based on the expression levels of a small number of genes, can be useful in the early and accurate diagnosis of MPM and lung cancer. There are 181 tissue samples (31 MPM and 150 ADCA). The training set contains 32 of them, 16 MPM and 16 ADCA. The remaining 149 samples are used for testing. Each sample is described by 12533 genes.

Central Nervous System Embryonic Tumour dataset

This classification task is based on DNA microarray gene expression data concerning central nervous system tumours (Pomeroy S L, *et al.* (2002)). *Survivors* are patients who are alive after treatment whiles the *failures* are those who succumbed to their disease. The dataset contains 60 patient samples, containing 21 medulloblastoma survivors and 39 treatment failures. There are 7129 genes in the dataset. The division of dataset is randomly done by us, with 30 in the training set and 30 in the test set.

Optical Digit and Ionosphere Datasets:

The Optical digit data (Kaynak C (1995)) is extracted normalised bitmaps of handwritten digits from a preprinted form by preprocessing programs. From a total of 43 people, 30 contributed to the training set and a different 13 to the test set. 32x32 bitmaps are divided into non-overlapping blocks of 4x4 and the numbers of on pixels are counted in each block. This generates an input matrix of 8x8 where each element is an integer in the range 0 to 16. This reduces dimensionality and gives invariance to small distortions. For each Attribute, all inputs attributes are integers in the range 0 to 16, and the last attribute is the class code: 0 to 9.

As the optical digit data has 3823 training samples and 1797 test samples with 10 classes, to simplify the classification problem and more concentrate on the algorithm itself, we reinterpreted the dataset to consider it as ten datasets, each providing a two class classification problem. In the first of these datasets, for example, all of the Opt Digit data is used, but the task is to classify a test sample as either class 0, or *not* class 0. In the second dataset, the task is to classify a sample as class 1, or *not* class 1, and so on. The 10 resulting cases are shown in Table 4.2 which shows the DCVs arising from these datasets.

DCV	**10 Classes**
0.567	0
0.451	1
0.560	2
0.388	3
0.513	4
0.381	5
0.512	6
0.663	7
0.280	8
0.378	9

Table 4.2. Optical recognition of handwritten digits.

First, choose class 0 (class from 0 to 9) as the predicted class 0, then incorporate the other classes 1 to 9 into another class 1. The entire process was repeated 10 times for different classes. Finally we acquire 10 new two-class datasets and the correlation values of each of them are showed in above table.

Ionosphere data is from Johns Hopkins University Ionosphere database. This radar data was collected by a system in the research by Sigillito V G, *et al.* (1989). The system consists of a phased array of 16 high-frequency antennas with a total transmitted power on the order of 6.4 kilowatts.

4.3.2 Data Correlation

As shown in Figure 4.4, the three groups of datasets have quite different ranges of correlation values. The difference between each type of data is very clear. The proteomics data often have high DCVs, which mean strong correlation with the target features. However, the DCV is sometimes very low, with the pancreatic dataset having a DCV of just 0.185 which is the second worst all over the datasets. In contrast the DCVs of OP and IO are mostly within a middle range. We also use the reduced ovarian datasets, as described in chapter 3, so that we examine a full range of DCV values.

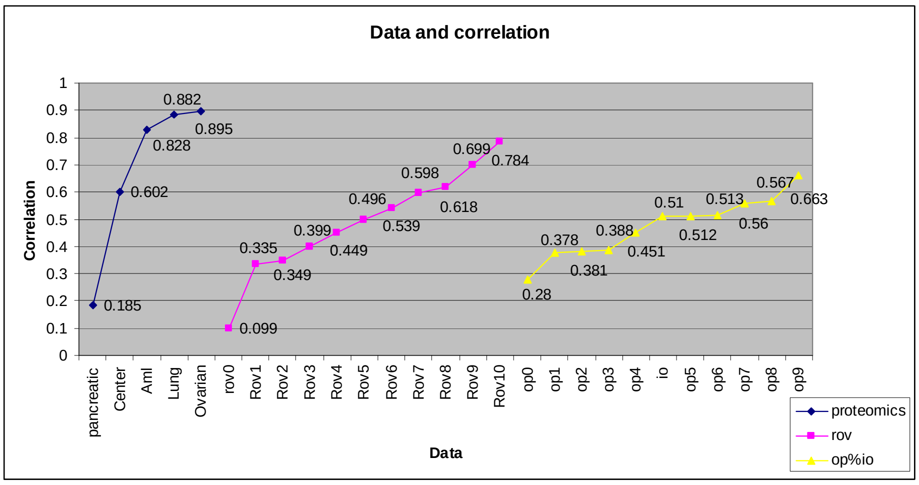


Figure 4.4. The DCVs of the three groups of datasets.

The first type of data is real proteomics data (blue curve); the pink curve exhibits the wide range of DCVs in the reduced ovarian data. The yellow curve comes from the optical digit data and ionosphere data, is which the attributes are limited to very small numbers.

4.4 Experiments Comparing FS and FB Strategies

4.4.1 Introduction

Classification accuracy on test sets is our primary measure used to compare the performance of FS and FB. We first applied the following three algorithms: CFB, RFB and EAFB, obtained by first applying feature selection and next by training a classifier on the full training set.

For each of CFB and RFB, a subset of *N* top features was obtained by the corresponding FS method, and the remaining features are placed in another subset. For the EAFB method, firstly a run of 500 iterations of the EA was done, and the *N* best features were chosen in the same way that these are chosen for EAFS; the remaining features, as before, are placed into another subset. In these experiments, *N* was always set to 100. In other words, we modified the *x* parameter in each case (see equation 1) according to the size of the dataset, to ensure that the number of pre-selected features was 100. This enables fair comparison with the FS methods.

The difference between FS and FB is basically that the evolutionary algorithm was run on the reduced dataset for FS methods, and the full dataset in FB methods. In all cases, the result of an individual experiment is the rule that performed best on the training set, with its performance measured on the test set. This was in every case averaged over five separate independent runs. The summary of test performance results is in Tables 4.3 to Table 4.5.

4.4.2 OP and IO Datasets

General notes for reading following tables are 1) Results from training datasets are *not* included; 2) Bold entries indicate the best performance for the dataset in that row; 3) The lowest rank stands for the best performance; 4) When the rank of a method is either 1st, 2nd or 3rd (1, 2, or 3), this is highlighted in the table by using, e.g. "*1*".

DCV	CFS Test	CFB Test	RFS Test	RFB Test	EAFS Test	EAFB Test
0.280	78.709	90.150	90.206	90.262	**90.306**	90.273
0.378	85.264	**89.945**	88.788	89.705	89.805	89.511
0.381	80.412	91.697	89.627	93.044	89.594	**93.122**
0.388	86.722	89.182	89.538	89.560	**89.683**	89.649
0.451	88.837	90.206	**90.762**	90.161	89.872	90.295
0.51	**91.523**	70.728	64.636	71.126	85.298	68.079
0.512	89.438	90.095	90.306	90.495	**90.929**	90.262
0.513	**94.224**	92.51	90.918	93.311	90.774	93.734
0.560	91.497	91.241	91.664	91.931	91.263	**92.354**
0.567	99.065	**99.154**	93.155	99.032	92.498	98.965
0.663	92.577	**93.767**	91.107	91.875	91.152	93.378

Table 4.3. The results from FS/FB strategies on OP and IO datasets.

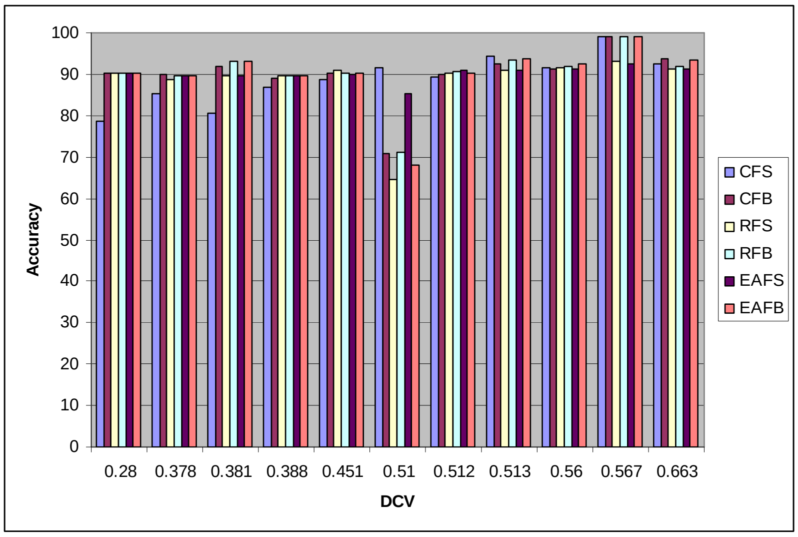


Figure 4.5. A graphic presentation of the results in Table 4.3. The largest differences between algorithm on the same dataset occur when the DCV is 0.51, which is the IO dataset. The other datasets are the OP datasets, which show similar performance differences among all algorithms. This is partly explained by the fact that the non-target attributes are of course all the same in these datasets.

As shown in Table 4.3, which compares the performance of FS and FB strategies on OP and IO datasets, an FS strategy takes 6 of the “best places” and FB strategies win 5 of the best places. Our interpretation of this is that both types of strategy can be seen as successful strategies for further research. The difference is, as yet, FS methods are already widely applied in many areas and FB methods are early in their application.

4.4.3 Reduced Ovarian Datasets

DCV	CFS Test	CFB Test	RFS Test	RFB Test	EAFS Test	EAFB Test
0.099	79.360	79.2	69.12	78.080	**80.16**	78.24
0.335	63.200	**64.16**	**64.16**	63.36	64.0	**64.16**
0.349	89.760	88.480	83.04	**90.720**	89.12	88.96
0.399	86.24	91.04	84.16	91.2	89.76	**92.0**
0.449	86.24	**92.64**	82.72	89.760	91.04	91.679
0.496	90.24	89.12	87.52	**93.76**	91.04	91.36
0.539	77.6	76.96	65.28	**78.56**	74.56	76.16
0.598	**93.2**	90.88	86.4	91.36	92.32	92.64
0.618	**84.96**	78.4	65.76	76.32	74.88	79.52
0.699	87.04	88.32	75.2	88.32	88.0	**89.44**
0.784	84.16	**86.08**	68.0	84.960	79.68	85.44

Table 4.4. The results from FS/FB strategies on reduced ovarian datasets. There are in total 11 datasets with DCVs ranging from 0.099 to 0.784.

On the reduced ovarian datasets, FB seems to be much more useful in the shown results than was the case for the OP and IO datasets. Both CFB and EAFB have three best places, appearing over the whole range of DCV values. This contrasts with CFS, for which the two best places only happen when the DCV is above 0.5.

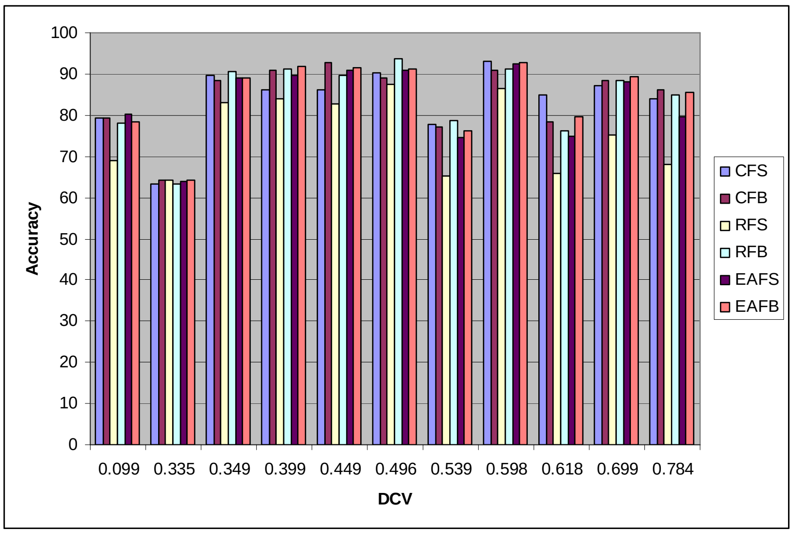


Figure 4.6. A graphic view of the results from Table 4.4. On the reduced ovarian datasets, the performances of RFS (the white column) are generally the worst overall, while the performance of EAFB (the pink column) are the best overall, however when the DCV is above 0.5, the algorithms involving statistical correlation (CFS and CFB) have strong performance.

4.4.4 Proteomics Datasets

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**
0.185	46.904	51.190	45.952	49.047	50.476	**53.571**
0.602	51.333	47.334	44.0	41.333	**60.667**	44.666
0.828	83.53	70.588	**87.059**	71.765	69.412	66.471
0.882	**95.168**	89.128	71.812	85.906	92.483	84.027
0896	**98.88**	93.12	83.68	91.84	88.96	93.28

Table 4.5. The results from various FS and FB strategies on real proteomics datasets. There are in total 5 datasets with DCV ranging from 0.185 to 0.896. Note the clear benefits of EAFS/EAFB for low DCVs, and the benefits of CFS for high DCVs.

On real proteomics datasets, CFS is still quite reliable for the datasets with DCV above 0.5. But, EAFB is clearly the best method for the dataset with the lowest DCV.

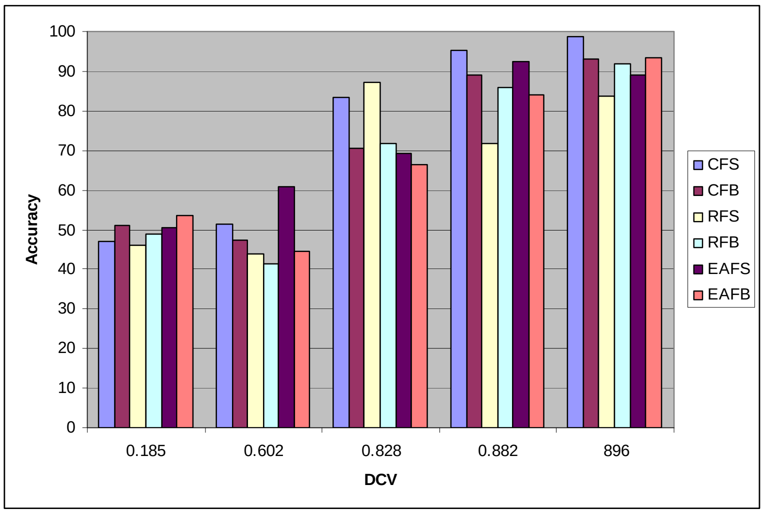


Figure 4.7. A graphic presentation of the results in Table 4.5. Notice that four of the five datasets have DCVs above 0.5, and it is therefore not surprising that CFS is best or second best for these four datasets.

4.5 Combined Analysis

4.5.1 Datasets with DCV below 0.5

Considering the results of the experiments summarised above, we note that FB often improves the performance of the basic FS methods, particularly in the cases of the datasets with DCV below 0.5. In attempt to gain a better understanding of the overall results, we look again at the results in terms of rank values. Also, we add further experiments so that we can compare with two other methods considered in the last chapter: CRFS and NFS. Eight methods were therefore applied to each dataset. In the tables and discussion below, an algorithm is given a rank of 1 if it achieves the best performance of the six methods on a particular dataset (or a group of datasets, depending on the context), and a rank of 8 means worst performance. Table 4.6 illustrates the ranks of each algorithm when we consider the whole group of datasets with DCV below 0.5.

DCV	CFS Test	CFB Test	RFS Test	RFB Test	EAFS Test	EAFB Test	CRFS Test	NFS Test	Dataset
0.099	*2*	*3*	7	6	*1*	5	7	4	rOV1
0.185	7	*3*	8	6	4	*2*	5	*1*	PA
0.335	8	*3*	*1*	6	4	*2*	4	6	rOV2
0.349	*3*	6	8	*2*	4	5	7	*1*	rOV3
0.399	7	*3*	8	*2*	6	*1*	4	5	rOV4
0.449	6	*1*	8	4	*3*	*2*	7	5	rOV5
0.496	5	6	8	*1*	4	*2*	*3*	7	rOV6
0.451	8	4	*1*	5	6	*3*	*2*	7	OP1
0.388	8	6	4	*3*	*1*	*2*	7	5	OP3
0.381	8	5	6	*3*	7	*1*	4	*1*	OP5
0.378	8	*2*	7	5	*3*	6	*1*	*3*	OP9
0.280	8	6	5	4	*2*	*3*	7	*1*	OP8
Total	78	43	73	48	45	**34**	58	46	

Table 4.6. Ranks of performance on the datasets with DCV below 0.5 per dataset for each FS/FB strategy.

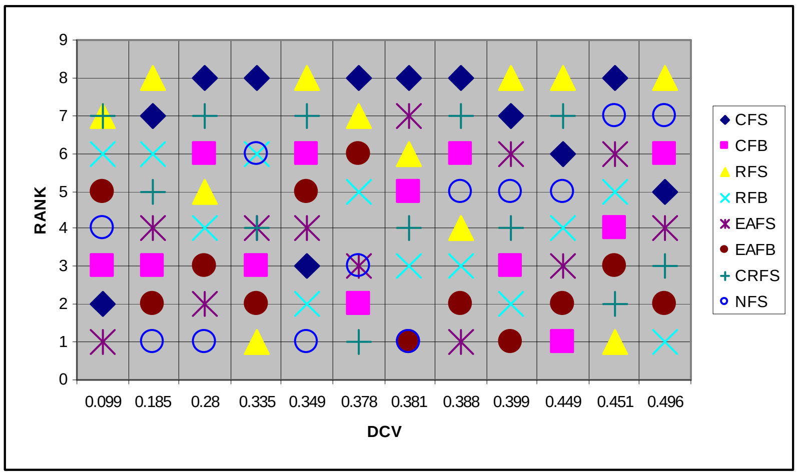


Figure 4.8. A graphic presentation of the ranks in Table 4.6. CFS never appears as the best ranked algorithm for these datasets. In fact, in 6 of the 12 cases, CFS is the worst-ranked algorithm.

In Table 4.6, the "Total" column can be seen as a measure of the overall rank over this group of datasets, with lower values being better. EAFB gets the best overall rank, and that suggests that, overall, EAFB is the best method. CFB and EAFS also seem very good overall. If we compare each FS method with its corresponding FB version, we can see that the FB version always has a better rank. It is notable, again, that CFS is never the best-ranked algorithm for these datasets with DCV below 0.5, and is often the worst-ranked algorithm. However, CFB has much better performance than CFS, appearing 6 times among the best three ranks. Overalls, the most successful method here is EAFB, appearing 7 times as rank 1 or rank 2.

4.5.2 Datasets with DCV over 0.5

For the datasets with DCV above 0.5, Table 4.7 summarises the results, involving again the eight methods tested in the previous section.

DCV	CFS Test	CFB Test	RFS Test	RFB Test	EAFS Test	EAFB Test	CRFS Test	NFS Test	Dataset
0.51	*1*	6	8	5	*3*	7	4	*2*	IO
0.539	*2*	*3*	7	*1*	6	4	8	5	rOV7
0.598	*1*	5	8	4	3	*2*	7	6	rOV8
0.618	*1*	*3*	7	5	6	*2*	8	4	rOV9
0.699	6	*3*	8	*3*	5	*2*	7	*1*	rOV10
0.784	4	*1*	8	*3*	6	*2*	7	5	rOV11
0896	*1*	*3*	7	5	6	*2*	8	4	OV
0.663	5	*1*	8	6	7	*3*	*2*	4	OP7
0.567	*3*	*2*	7	4	8	5	*1*	6	OP0
0.560	6	8	5	*3*	7	*1*	4	*2*	OP2
0.513	*2*	6	7	5	8	*3*	*1*	4	OP4
0.512	8	6	4	*2*	*1*	5	7	*3*	OP6
0.602	*1*	4	8	5	*2*	7	*3*	6	CNS
0.828	*2*	5	*1*	4	6	7	*3*	8	Aml
0.882	*2*	*3*	5	7	*1*	4	6	8	Lung
Total	**45**	59	98	62	75	56	76	68	

Table 4.7. Ranks of performance (1=best) per dataset for each FS/FB strategy, only for datasets with DCV above 0.5.

When observing the results in Table 4.7, and consider the ‘Total’ row, which gives an idea of the overall performance of the method in each column, we discover that a Feature Bias strategy improves on the basic FS approach in just two cases now (RFB vs RFS and EAFB vs EAFS), EAFB is no longer the best overall strategy, but it still performs well, and seems to be the second-best overall strategy for these datasets.

CFS is certainly the best overall method for the higher DCV datasets, but we note that CFB does perform better than CFS in five of the fifteen cases. Meanwhile, if we look at DCVs above 0.6, most of the CFS and CFB ranks are among the top three.

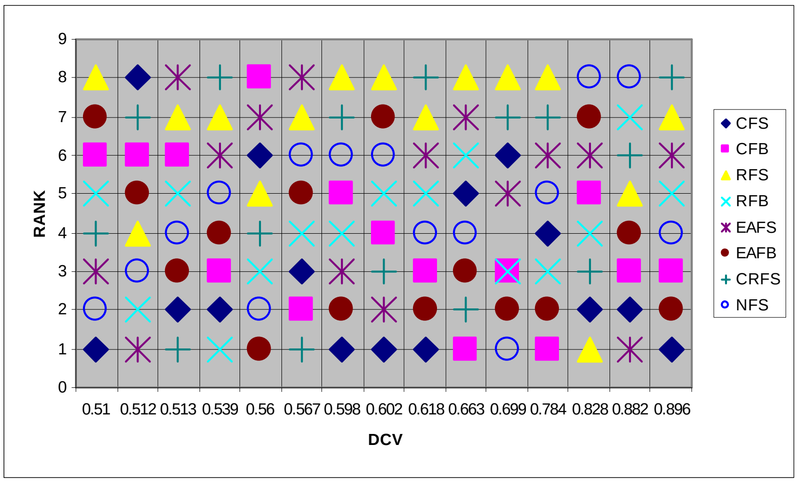


Figure 4.9. A graphic presentation of the ranks in Table 4.7. When the dataset DCV is above 0.5, the best algorithm overall is not EAFB or NFS (which performed well for datasets with DCV below 0.5), although these methods still show reasonable performance. Instead, CFS, which was the worst algorithm for DCV below 0.5, is clearly the best algorithm overall for these datasets. CFB also shows strong performance, especially when the DCV is above 0.6.

The results suggest that, for this collection of datasets, there is a relationship between the DCV and the best choice of Feature Management (FM) approach. So, this adds to evidence gathered in Chapter 3, to suggest that it seems justifiable, as a first step while faced with a new dataset, to check the DCV of the dataset, and use it as a guide towards the appropriate FM strategy. Broadly speaking, if the DCV is rather high, CFS and CFB could be considered the best choices (from among the methods we have examined). However, EAFB would generally be a good choice in most conditions, especially when the DCV is low. If for some reason the DCV is not known in advance, then we suggest that EAFB is the best choice. The FB strategies seem very competitive when compared with the traditional feature selection methods. EAFB in particular seems to deserve more study.

4.5.3 Additional Experiments with CFS/CFB and RFS/RFB, Relief F

Further investigations were done to examine some basic versions of combined FS/FB strategies, and some basic parameter variation of the Relief F method. In a combined FS/FB strategy, the idea is to do FS first, but to select 1,000 features rather than 100. FB then operates as normal on the 1,000-features reduced dataset. This leads to much improved computational processing time for the machine learning, which is a benefit from a pure FS method, and retains the benefit of a pure FB method since a much larger collection of features is retained.

The experiments with RFS (10/3) are motivated by the fairly poor performance of Relief F that we have seen so far in experiments, and by considering Khayat *et al.* (2008) which proposes a more stable Relief F method. It suggests that, the choice of the parameter *k* for the *k*-NN classifier built into the Relief F method is sensitive to how many attributes are included. As a basic exploration of this, we try Relief F with *k* = 3 and *k* = 10.

These experiments are done for the proteomics datasets only, and the results are shown in Table 4.8. The table also includes the previous comparable results for the other algorithms tested in this chapter, so that we can compare easily.

CFS again shows strong performance, with joint-best overall rank on the proteomics datasets, which is to be expected since they are mostly with high DCVs. The method that joins CFS in best overall rank is CFS/CFB. However CFS is only the best-ranking method for one of the datasets, and the same is true for CFS/CFB. There is a tentative argument to suggest that CFS/CFB is more robust than CFS, since CFS ranks 11th out of 12 in one case (with the lowest DCV), while CFS/CFB never performs worse than 7th out of 12.

DCV	Dataset	CFS Test	CFB Test	RFS Test (10 / 3)	RFB Test (10 / 3)	EAFS Test	EAFB Test	CRFS Test	NFS Test	CFS/CFB RFS/RFB
0.185	PA	11	*3*	12 / 10	9 / 7	5	*2*	6	*1*	7 - 4
0.602	CNS	4	5	7 / *2*	10 / 11	*1*	6	9	12	*3* - 7
0.828	AML/ ALL	*2*	9	*1* / *3*	8 / 5	10	11	6	12	7 - 4
0.882	LUNG	*1*	6	12 / 10	7 / 5	*3*	9	4	8	*2* - 11
0.896	OV	*2*	4	11 /10	6 / 6	9	*3*	12	5	*1* - 8
	Total	20	27	43 /35	40 / 34	28	31	35	38	20 - 34
Above 0.5	Total	**9**	24	31 / 25	31 / 27	23	29	29	37	13 - 30

Table 4.8. The performance of twelve different FS, FB and combined methods on proteomics datasets. Three new methods are introduced to compare with results that have been considered in other tables: relief F with *k* = 3, CFS/CFB, and RFS/RFB.

Relief F with *k* = 3 does achieve better overall ranks than with *k* = 10, in both the RFS and RFB versions (and the combined RFS/RFB strategy used *k* = 3 in this case). However Relief F is still outperformed by EAFS and EAFB. When we look only at the cases with DCV > 0.5 (which means we just omit the pancreatic dataset), the preference for CFS and CFS/CFB is very clear, with EAFS still providing a robust performance with the next-best overall rank.

4.6 Conclusions and Discussions

In the chapter, we described the idea of Feature Bias (FB) strategies and investigated the performance of three FB strategies in comparison with the performance of the corresponding Feature Selection (FS) strategies. The three basic strategies compared were standard correlation-based FS / FB, Relief-F based FS / FB and EA-based FS / FB (in which an initial short EA run is used to identify good features). The comparisons were done over a collection of (mainly) many-attribute datasets, largely using proteomics data. Our investigation also considered the dataset correlation value (DCV) of each dataset – this is simply the largest value, over all features in the dataset, of the statistical correlation between a feature and the target attribute in that dataset. Some of the datasets were generated by adding noise to the Ovarian data, so that we could obtain datasets with a wide spread of DCV values. By looking at mean performance on the test sets, we can claim that the FB strategies tend to outperform the corresponding FS strategies. In particular, EAFB has the strongest overall performance among these strategies and considering all datasets tested that had low DCV, suggesting that it is unwise to put too much trust in basic statistical correlation measures for feature selection, especially when the dataset in question has a low DCV. However, with high DCV, CFS certainly appears to be the best strategy. Naturally these findings are for a particular collection of datasets, but the suggestion is that these findings may be more generally true. Future work is warranted to further explore the FB strategies and also the relationship between performance and DCV.

We also explored combined FS/FB strategies, in which FS is used first, but it is less selective and still leaves the dataset with a large number of features. These explorations were focused on the proteomics datasets (which tend to have high DCV) and we found that CFS, CFB and CFS/CFB dominated the results. We also found that Relief F performance was quite affected by variation in *k*, but the improvement was not able to compete with the overall better methods on the datasets tested.

The drawback of a Feature Bias strategy is that, since no features are eliminated, we lose the potential speedup that arises when FS strategies are used. However, in any particular application this level of speedup may not be important, either because the computation time involved is acceptable in context, or because the importance of accurate results outweighs such issues. Meanwhile, it is clear that there is a continuum between pure FS and FB strategies which can be explored to find ideal speed/quality tradeoffs. For example, FS could be used to reduce an enormous dataset to one that is more manageable, but still has so many features that further FS would normally be used; however FB, rather than FS, would then be applied to minimise the potential damage of removing too many relevant features.

Chapter 5

Further Investigation of Relationship between DCV and Feature Selection Performance

5.1 Introduction

The problem of selecting relevant features has been the focus of interest for complex machine learning and data mining tasks. Feature selection (Singhi S and Liu H (2006)) allows for faster model building by reducing the number of features, but also helps remove irrelevant, redundant and noisy features, this in turn allows for building simpler and more comprehensible classification models with good classification accuracy. Much work has been done and many state of the art algorithms are proposed to improve the performance for many attribute datasets. Pearson's correlation coefficient, one of simplest and widely used FS method, proves to be usually very effective strategy while applying in a filter model, even though it does not remove feature redundancy and therefore yields unnecessarily large feature subsets. However, when these correlation values tend to be rather low for all features (common in many datasets of importance), the basis for pre-selection of any specific set of features is undermined, and straightforward feature selection may do more harm than good.

Through the observation of experiments, EAFB has the strongest overall performance, suggesting that it is unwise to put much trust in basic statistical correlation measures for feature selection, especially when the dataset in question has a low DCV. However, with highly correlated data, CFS certainly appears to be the best strategy. Naturally these findings are only for a particular collection of datasets, however the suggestion is that these findings may be more generally true. An interesting phenomenon is that FS performance seems related to the DCV of a dataset. The DCV may therefore be able to play an important role for choosing the correct FS strategy in advance. In this chapter, we will analyse the results more closely to see the significance of this effect.

After considering the results of the various experiments, we investigate the relationship between the performance of the methods and the DCV by testing for correlations between these two variables; we use either Pearson's correlation coefficient or Spearman's correlation coefficient, depending on the analysis being done. Spearman' correlation is nonparametric, so it tests the correlation between the rank-ordering of performance and the rank-ordering of DCV. This allows us to see how the relative performance of the different FS methods varies according to DCV. E.g. we look at this question for each method X: "is it true that when the DCV is low (high) the relative performance of method X tends to be low (high)?". We also ask another question that can be simply expressed as follows, for techniques X that are not CFS: "is it true that when the DCV is low (high), method X is likely to be better (worse) than CFS? For that question, we interpret the data to provide a simple value (0, 1 or 2) concerning the relative performance of X against CFS, and use Pearson's correlation coefficient to examine the correlation between this value and DCV.

In the rest of this chapter, we will first cover some basic background on correlation in section 5.2, and then look at the work that tries to answer the questions considered above in sections 5.3 and 5.4. After that, in section 5.5 we try to draw a blueprint for guiding the choice of FS method for different types of datasets. Section 5.6 discusses conclusions and future work.

5.2 What is Correlation?

A correlation coefficient tells us whether two variables vary together. The most common tests for correlation are the Pearson product-moment correlation coefficient and the Spearman rank-order correlation coefficient. Both vary from −1 (perfect negative correlation) through 0 (no correlation) to +1 (perfect positive correlation)[3].

As a general "rule of thumb," correlation is often viewed along the following continuum **(**MacFarland T W (1998)):

1) 0.00 to +0.39 = no positive correlation between *x* and *y*.
2) 0.00 to −0.39 = no negative correlation between *x* and *y*.
3) +0.40 to +0.79 = mild positive correlation between *x* and *y*.
4) −0.40 to −0.79 = mild negative correlation between *x* and *y*.
5) +0.80 to +0.99 = strong positive correlation between *x* and *y*.
6) −0.80 to −0.99 = strong negative correlation between *x* and *y*.

Also, the size of sample *n* is necessary to consider when the significance of a correlation was not too obvious. Taken the example, if *n* = 15, then the correlation of −0.413 could be seen as significant, however, it is not when *n* is below 10.

A high correlation between variables *x* and y does not imply that one variable is *causing* the other, it simply means that these two variables are related in some way. There are many reasons why variable *x* and y could be highly correlated. A high correlation could be the result of (a) *x* causing y, or (b) *y* causing *x*, or (c) a third *z* causing both *x* and *y*, or (d) many more variables being involved. The only method that can strictly be used to infer cause are experimental methods where one variable is manipulated by the research, a second variable is subsequently observed, and all other variables are controlled.

---

3 Note that the negative (− or decrease) and positive (+ or increase) signs in correlation are only used to suggest direction. The negative sign does not mean "bad" and the positive sign does not mean "good".

Pearson's correlation coefficient is calculated based on the actual values of the variables. In contrast, Spearman's correlation coefficient is based on the rank ordering of the variables. Spearman's correlation can be used when there is non-parametric data, or when parametric data is considered after being processed into ranks. In fact, of course these correlation measures play a part in common feature selection strategies. Both CFS and CFB select the features that have highest (Pearson's) correlation with the target class.

5.3 Predictability of Performance from DCV

5.3.1 Introduction of Spearman's Correlation

Spearman's rank correlation coefficient (Lyerly (1952)) is appropriate when both variables are ordinal. Charles Spearman, the famous quantitative psychologist, developed this type of correlation. For both variables, either the data are already available in ranks, or the researcher converts the raw data to ranks before the analysis. Spearman's Rank-Difference Coefficient of Correlation (sometimes called "rank difference coefficient after one method of calculating it") is viewed as the non-parametric test for determining if there is an association between phenomena.

The equation for calculating the Spearman's rank correlation is:

$$p = 1 - \frac{6\sum d^2}{(n^3 - n)}$$

Where *p* denotes the population Spearman correlation and d represents the difference between the ranks on variables *x* and *y* for individual *i*, *n* denotes the number of ranks.

Full details are in the article by Lamb G S (1984). The formula is only strictly appropriate when there are no ties among the ranks for either variable. If the number of ties goes too high, the formula given is only approximate.

The simple procedure for calculating the coefficient is:

- Collect the data for the two variables, e.g. as two columns of values.
- Convert the numbers into ranks. Ranking is achieved by giving the ranking '1' to the biggest number in a column, '2' to the second biggest value and so on. The smallest value in the column will get the lowest ranking. This should be done for both sets of measurements.
- Tied scores are given lower rank value. For example, if we rank the numbers: 6, 7, 8, 8, 20; the ranks are 5, 4, 2, 2, 1. The two 8s are considered as joint 2nd, rather than joint 3rd.
- Find the difference in the ranks (*d*): This is the difference between the ranks of the two values on each row of the table.
- Square the differences (d^2) To remove negative values and then sum them ($\sum d^2$)

5.3.2 The Link between DCV and Algorithm Performance: Reduced Ovarian Datasets

In this section, we will investigate the link between DCV and the performance of each algorithm according to Spearman's correlation. First, on the reduced datasets, we test the interaction between DCV and the performance of one algorithm: CFS as an example, then we will present all the links and give an analysis about the experiments.

The link between DCV and CFS and other algorithms

Let the variables *x*, *y* be given below (*x* represents dataset DCV, and *y* represents average accuracies from an algorithm). While $x = 0.349$ and $y = 89.76$ means the following: for the dataset with DCV= 0.349, the accuracy of machine learning experiments on test sets was found to be 89.76%. Table 5.1 illustrates this for CFS, where the accuracies are the mean of five test set accuracies on the reduced ovarian datasets.

x_i (DCV)	0.099	0.035	0.349	0.399	0.449	0.496	0.539	0.598	0.618	0.699	0.784
y_i (CFS)	79.36	63.2	89.76	86.24	86.24	90.24	77.6	93.2	84.96	87.04	84.16

Table 5.1. Data from our previous experiment. It includes DCV of rOV datasets, and average accuracies from CFS.

First we convert the accuracies to ranks, where the highest raw score received a rank of 1. Now replace each x_i by its rank value, and similarly for y_i. For the current example case, this leads to Table 5.2.

x_i (DCV)	11	10	9	8	7	6	5	4	3	2	1
y_i (CFS)	9	11	3	5	5	2	10	1	7	4	8

Table 5.2. The data from Table 5.1 is expressed here as ranks. For example, the highest DCV value (1) is associated with the 8th best (out of 11) performance from CFS on these datasets.

Recall that we calculate Spearman's rank correlation coefficient for one algorithm as follows, here showing the results for the current example of CFS and reduced ovarian datasets:

$$\sum d^2 = abs(y_i - x_i) = 269 \tag{1}$$

$$n^3 - n = 1320 \tag{2}$$

$$p_{DCV} = 1 - \frac{6\Sigma d^2}{(n^3 - n)} = 0.22 \tag{3}$$

When we do the same, for each of a number of different FS methods on the reduced ovarian datasets, we get the results shown in Table 5.3. For example, in this table, we find that, when measured by performance on the reduced ovarian datasets, the correlation level between DCV and the performance of EAFS is 0.21.

	CFS	CFB	RFS10	RFS3	RFB10	RFB3
p_{DCV}	0.222	0.018	0.027	0	0.1	0
	EAFS	EAFB	CRFS	NFS	CFS/CFB	RFS/RFB
p_{DCV}	0.041	0.21	0.105	0.141	0.255	0.068

Table 5.3. Spearman's correlation between DCV and performance on 12 algorithms.

As shown in this table, on reduced ovarian datasets, every algorithm has no strong significance in its performance correlated with DCV. Almost all cases show positive value, so that the tendency is for a correlation that suggests higher DCV value means better performance. But in all cases the value is below 0.4, so that we cannot even infer a mildly significant correlation. This is quite different from our expectation, but as we shall see this is not the same for other groups of datasets. In the case of the reduced ovarian datasets, they were of course made to have different values of DCV to by adding different amounts of noise, and this means they may not be really reflective of real datasets. Before looking at other datasets, we first consider an alternative measure to DCV. Of course, DCV is a very simple summary of a dataset, because it presents only the highest value of any correlation between an attribute and the target attribute. This does not look at the distribution of correlations. We therefore explored a simple alternative called the mean data correlation value (MDCV). This is the mean value, over all attributes, of the correlation between that attribute and the target class.

The Link between MDCV and CFS on Reduced Ovarian Datasets

DCV	0.099	0.035	0.349	0.399	0.449	0.496	0.539	0.598	0.618	0.699	0.784
MDCV	0.047	0.08	0.205	0.229	0.24	0.245	0.205	0.256	0.288	0.253	0.388
CFS	79.36	63.2	89.76	86.24	86.24	90.24	77.6	93.2	84.96	87.04	84.16

Table 5.4. The MDCV and the DCV of the reduced ovarian datasets, shown with the mean performance on test sets of CFS. The rank ordering of MDCV will be different from the rank ordering of DCV. This is now shown in Table 5.5.

For the reduced ovarian datasets, we show Table 5.4 which gives the MDCV together with the DCV so they can be compared.

DCV	11	10	9	8	7	6	5	4	3	2	1
MDCV	11	10	8	7	6	5	8	3	2	4	1
CFS	9	11	3	5	5	2	10	1	7	4	8

Table 5.5. This is the data in Table 5.4 now converted to ranks.

After repeating the processes described above, now considering the correlation between MDCV and performance, we calculate the *p* values for 12 algorithms and show the results in Table 5.6.

	CFS	CFB	RFS10	RFS3	RFB10	RFB3
p_{MDCV}	0.445	0	0.231	0.222	0.277	0.222
	EAFS	EAFB	CRFS	NFS	CFS/CFB	RFS/RFB
p_{MDCV}	0.318	0.468	0.109	0.272	0.454	0.336

Table 5.6. Spearman's correlation values for relationship between MDCV and performance on the reduced ovarian datasets. There now is weak significance shown for CFS and EAFB and CFS/CFB.

Now we have the MDCV, this seems to show some slight difference. The MDCV carries more information about the correlations within a dataset, and this leads to showing some mild significance (above 0.4) in the case of CFS, EAFB and CFS/CFB. This is some small support for the idea that a simple summary statistic of the dataset can provide predictive idea of FS algorithm performance, but the significance is not strong. We return to a reminder that the reduced ovarian datasets may not be indicative of real datasets. Adding noise for each attribute in the ovarian data, and also limiting the datasets to 1,000 attributes produced the reduced ovarian datasets. As we know, the ovarian data has 15,154 attributes, and the noise addition means that, especially as the DCV/MDCV becomes lower, the datasets are less and less like realistic ones. Next we look at real-world datasets.

The Link between DCV and Algorithm Performance: Real Datasets

Here we directly give the results on five real (proteomics) datasets (cutting out the details of calculation). These results appear in Table 5.7.

	CFS	CFB	RFS10	RFS3	RFB10	RFB3
p_{DCV}	1	0.9	0.6	0.9	0.9	0.9
p_{MDCV}	0.7	0.6	0.6	0.6	0.6	0.6
	EAFS	EAFB	CRFS	NFS	CFS/CFB	RFS/RFB
p_{DCV}	0.9	0.9	0.8	0.9	1	0.8
p_{MDCV}	0.9	0.6	0.8	0.6	0.7	0.5

Table 5.7. The Spearman's correlation values on five proteomics datasets, calculated for each of 12 FS algorithms, showing the correlation of DCV with performance, and also of MDCV with performance.

From Table 5.17, the lowest *p* value is 0.5. This means that all correlations here are positive (higher DCV or MDCV associates with better performance), and they are all at least mildly significant. The DCV seems to be clearly the better predictor, which leads to *p* values always at least as high as the values from MDCV and often higher. Considering the DCV, in all cases (except only for RFS with *k* = 10), a strong correlation is shown between DCV and the performance of the algorithm. When we look at the real proteomics datasets, we can therefore see evidence that the DCV can be a good predictor of algorithm performance. However we must also recognise that the number of datasets is only 5. But, again, it is supporting evidence that we see the strong evidence of correlation in almost all of the 12 algorithm cases.

5.3.3 Analysis

Spearman's correlation has been is use for half century among scientists. The results from it could be seen as convincing enough for further discussion and research. It is clear that on the datasets produced by us (the reduced ovarian datasets), the significance shown by Spearman's coefficient is not strong. As discussed above, the possible reasons for this come from the way we added noise in these datasets, and so influencing the relationships in the data. Thus we could not claim significance on these "fake" datasets. However, even in these cases, 10 of the 12 methods showed a weak positive correlation, which is much more than can be expected by chance. In the next main section (5.4) we will use Pearson's correlation coefficient to investigate the relative performance of different FS algorithms. From our experience with the reduced ovarian datasets in this section, we expect it will not be useful to investigate them in the next section. So, we will only investigate Pearson's correlation for real datasets.

To the contrary, on the original proteomics datasets, there is clear evidence of relationship between DCV and the FS method performance. Even if could not entirely understand how the DCV affects the performance, we can conclude there is a relationship that has been found. This makes it possible to wonder if the likely performance of FS algorithms could be determined in advance by the DCV of the dataset. Of course, the performance of one algorithm could be limited into a certain range, and it is easy to imagine why CFS always performs very well on highly correlated data. However, CFS is not a stable algorithm on the poorly correlated datasets, and the correlations show that all of the algorithms perform less well when the DCV is low. The important thing, however, is the relative performance between different algorithms. Maybe some algorithms will tend to be better than CFS when the DCV is low. In the next section, we will use Pearson’s correlation coefficient to look at this question of the relative performance of algorithms and how it relates to DCV.

5.4 Predicting Whether or not a Method may CFS

5.4.1 The Pearson Product-moment Correlation Coefficient

The Pearson product-moment correlation coefficient value is a number between −1 and 1. It measures the strength and the direction of correlation between two variables such as a collection of pairs (*f*, *p*), and is calculated as below.

$$c_{f,p} = \frac{n\sum_{i=1}^{n} f_i p_i - \left(\sum_{i=1}^{n} f_i\right)\left(\sum_{i=1}^{n} p_i\right)}{\sqrt{\left(n\sum_{i=1}^{n} {f_i}^2 - \left(n\sum_{i=1}^{n} f_i\right)^2\right)\left(n\sum_{i=1}^{n} {p_i}^2 - \left(n\sum_{i=1}^{n} p_i\right)^2\right)}}$$

If *f* and *p* have a strong positive linear correlation, *c* is close to +1. In contrast, if *c* is close to −1, it would be towards a perfect negative correlation. If there is no linear correlation or only a weak linear correlation, *c* is close to 0.

If the correlation is strong (usually we define correlation value over 0.8 or below −0.8 as strong, and between −0.5 and 0.5 as weak), that means a high (positive or negative) correlation. In this section we use this correlation coefficient to test the correlation between: (i) the DCV of a dataset, and (ii) a number, 0, 1, or 2, that sums up the relative performance of CFS and another algorithm on that dataset.

5.4.2 The Link between CFS and the Performances of Other Algorithms on Real Data

DCV	CFS Test	CFB Test	RFS Test(10/3)	RFB Test(10/3)	EAFS Test	EAFB Test	CRFS Test	NFS Test	CFS/CFB RFS/RFB
0.185	46.9	51.1	45.9 / 48.5	49.0 / 50.0	50.4	53.5	50.2	**56.9**	50.0 / 50.9
0.602	51.3	47.3	44.0 / 54	41.3 / 40	**60.6**	44.6	42.0	39.3	52.0 / 44.0
0.828	83.5	70.5	**87.0** **/** 82.3	71.7 / 77.0	69.4	66.4	73.5	65.2	72.3 / 79.4
0.882	**95.1**	89.1	71.8 / 81.3	85.9 / 91.0	92.4	84.0	91.8	85.5	92.6 / 75.0
0.896	98.8	93.1	83.6 / 86.8	91.8 / 91.8	88.9	93.2	81.0	92.1	**99.0** **/** 90

Table 5.8. Original test set performance from 12 algorithms on proteomics datasets. The bold figure gives the best performance from any algorithm on the dataset in that row.

As we found previously in this chapter, examination of results on the reduced ovarian datasets shows that they are probably not useful for understanding what the situation is with real datasets. So in this experiment, we only do tests on the real datasets. The procedure includes three parts:

First, of course, we assemble all of the relevant data. This is summarised in Table 5.8. This table provides the performance (test set accuracy averaged over 5 runs) of each of the 12 algorithms we have been considering in this chapter. In this case we are looking only at the five real proteomics datasets.

Next we convert the data into values that show the relative performance against CFS. If the performance of an algorithm is better than CFS, we represent this with the value 2. If it is worse, we represent this with the value 0. If the difference is small (less than 0.1%) we represent this with the value 1. Table 5.9 shows these relative-to-CFS performance values of the 11 (non-CFS) algorithms based on the data of Table 5.8. In fact, it turns out in the case of the proteomics datasets that the performances compared with CFS were always more than 0.1% different, so Table 5.9 only contains values 0 and 2.

DCV	Rank	CFB	RFS10	RFS3	RFB10	RFB3	
0.185	1	2	0	2	2	2	
0.602	2	0	0	2	0	0	
0.828	3	0	2	0	0	0	
0.882	4	0	0	0	0	0	
0.896	5	0	0	0	0	0	
DCV	Rank	EAFS	EAFB	CRFS	NFS	CFS/CFB	RFS/RFB
0.185	1	2	2	2	2	2	2
0.602	2	2	0	0	0	2	0
0.828	3	0	0	0	0	0	0
0.882	4	0	0	0	0	0	0
0.896	5	0	0	0	0	2	0

Table 5.9. Representing the performance data in Table 5.8 by 0, 1, or 2, depending on whether an algorithm performed worse, similar, or better than CFS on the proteomics datasets.

Next, we calculate the Pearson's correlation coefficient between the DCV and the 0/1/2 performance number. Notice that we cannot use Spearman's coefficient here, because if we looked at the performance numbers in terms of ranks, there would be too many ties. In fact we do this in two ways to get two different Pearson's coefficients. In one case, we represent the DCV as the actual DCV value; in the other case we represent the DCV by its rank value. These correlation values are all shown in Table 5.10.

As Table 5.10 shows, there are many correlation values that seem to be strong. In all cases except for RFS10, there seems to be a clear negative correlation between the DCV (or the rank of DCV) and the relative-to-CFS performance indicator. That is, the lower the DCV, the more confident we are that the algorithm will be better than CFS; the higher the DCV, the more confident we are that the algorithm will be worse than CFS. Also, the values seem to be more significant when they are generated by the rank of the DCV values than by the DCV values themselves.

	CFB	RFS10	RFS3	RFB10	RFB3	NFS
DCV	−0.707	0	−0.866	−0.707	−0.707	−0.707
Rank	−0.919	0.278	−0.867	−0.919	−0.919	−0.919
	EAFS	EAFB	CRFS	CFSB	RFSB	
DCV	−0.866	−0.707	−0.707	−0.289	−0.707	
Rank	−0.867	−0.919	−0.919	−0.536	−0.919	

Table 5.10. For each of the 11 non-CFS methods, this table shows the correlation between the dataset DCV (or the rank of DCV) and the relative performance of the method and CFS (0, 1 or 2).

However, there are only 5 datasets involved here, and we need to be more careful about what values are significant. To investigate this, we notice that there are only 32 possible values of the correlation coefficient in each case (based on DCV or based on rank of DCV). This is because there are only 5 datasets, and only 2 performance values (0 and 2). So, there are precisely 32 different possible ways to distribute the performance values against the datasets. Table 5.11 shows these values. It shows 16 of the values in each case. The other 16 are the same set of values multiplied by −1.

DCV	DCV	Rank	Rank
0.0	0.354	0.143	0.0
0.707	0.289	0.214	0.405
0.354	0.0	0.193	0.379
0.866	0.577	0523	0.64
0.0	0.289	0.111	0.278
0.577	0.289	0.441	0.558
0.289	0.0	0.42	0.536
0.866	0.707	0.919	0.867

Table 5.11. The 16 different absolute values that can be obtained, when finding the correlation between DCV (or rank of DCV) and the performance-against-CFS indicator (either 0 or 2) on five datasets.

When we look at Table 5.11 and also consider the negative values, we can make the following notes. When the correlation is based on DCV, there are only 2 cases (out of 32) where the value is higher than 0.8. This is below 7%. So we can say that, if the value is above 0.8 (or below −0.8) we can be more than 90% sure that the value could not have been obtained by chance, and so there is a real positive (or negative) correlation. For other values we would have to conclude that significance is not really demonstrated. For values above 0.7, for example, we can have only 75% confidence. This shows a weak significance, but is not enough for a firm conclusion. When we look at the correlation values that are based on rank of DCV, the situation is very similar. If the value is above 0.8 (or below −0.8) we can conclude a positive (or negative) correlation.

We illustrate this graphically in Figure 5.1. It shows the absolute values available for the correlation values (in Table 5.11), each reported two times for its positive and negative form. What we are saying in the above text is that the four highest values on the right at each plot are the significant ones.

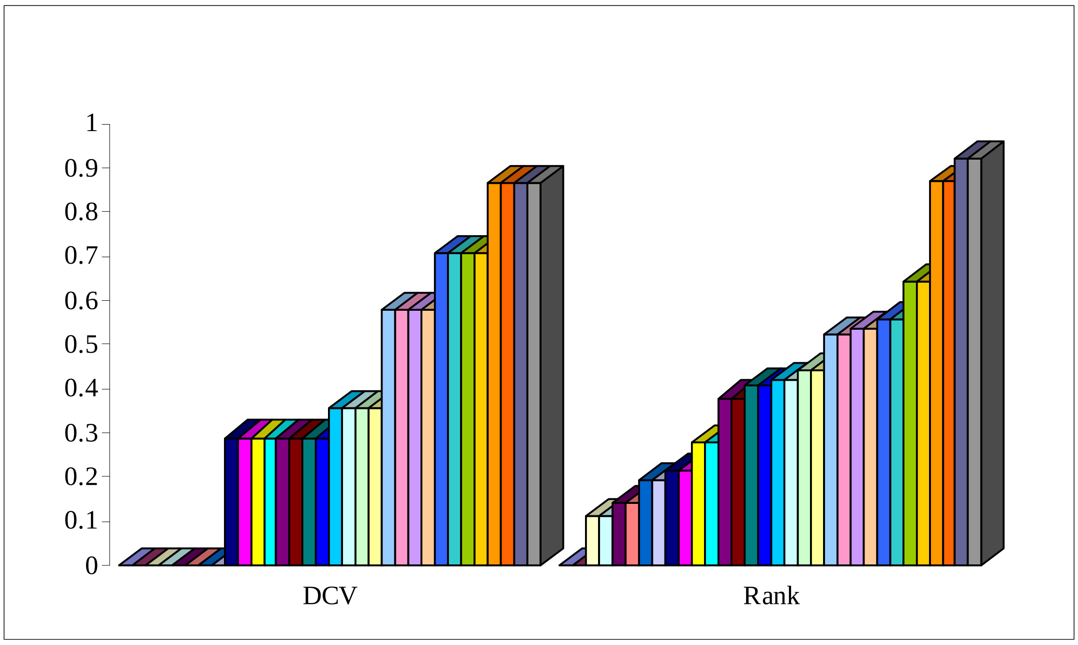


Figure 5.1. The distribution of all the possible correlation values, when there are only 5 datasets and two performance indicators (two values for the second variable). On the left are the values that come from using DCV as the first variable, on the right are the values that come from using the rank of DCV (that is, just the numbers 1 to 5) for the first variable.

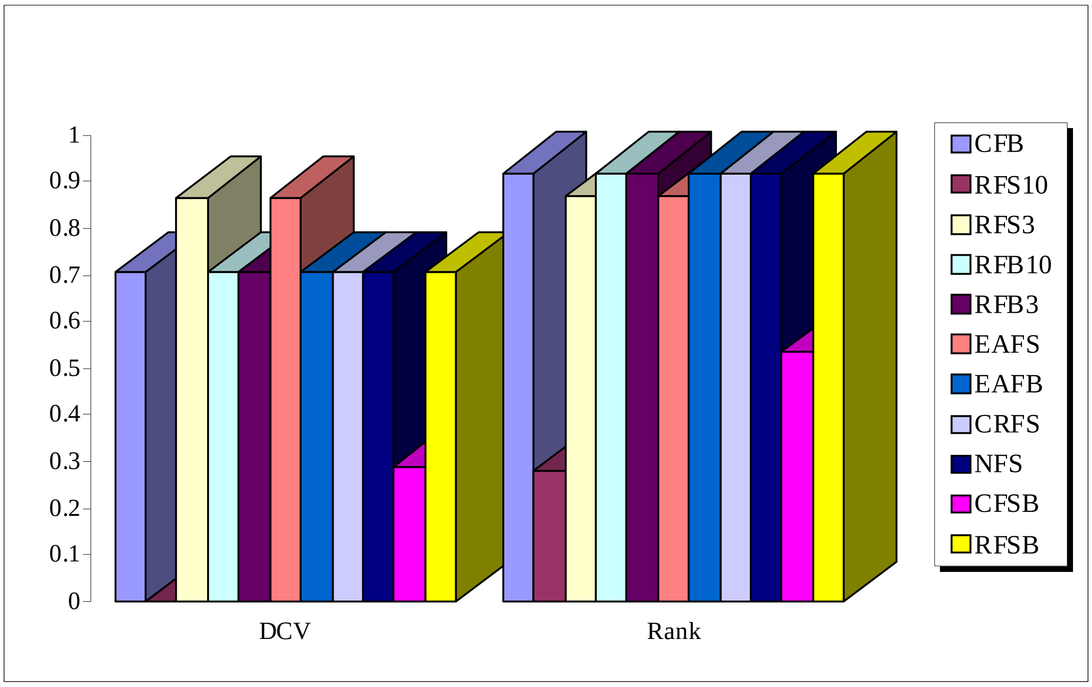


Figure 5.2. The distribution of all the possible correlation values, when there are only 5 datasets and two performance indicators (two values for the second variable). On the left are the values that come from using DCV as the first variable, on the right are the values that come from using the rank of DCV (that is, just the numbers 1 to 5) for the first variable.

In Figure 5.2, which is to the same scale as Figure 5.1, we can see the correlation values that we saw in Table 5.11. This helps to illustrate the general level of significance in Table 5.11.

Therefore we can see a general tendency towards the significance of the findings. A clear statement that we can make, for example, are that the performance of EAFS vs CFS is strongly negatively correlated with DCV; that is, as the DCV reduces, the EAFS is more likely to outperform CFS. The same is true for RFS with k = 3.

5.4.3 Interim Summary

From all tables above, we should conclude that, for the reduced ovarian datasets, there is no significance in the relationship between the DCV (or the MDCV) and the performance of any of the FS algorithms we have tested. However, there are different conclusions when we look at real datasets. In all cases, except for RFS with k = 10, when we use the DCV, we find that higher DCV means higher performance of the algorithm. That is, there is a positive correlation. Also, when we look at real datasets and at the different question of the correlation between DCV and the relative performance of an algorithm against CFS, we again find significant results. Especially when we base correlation on the rank of DCV, we can see a strong negative correlation between DCV and the chance that CFS is better than another FS method. In other words, this is the same as a strong positive correlation between DCV and the chance that another FS method is better than CFS. This is most clearly true when the other algorithm is EAFS.

In the rest of the chapter we look more into how we might use the DCV as a guide to choosing a good FS method for a given dataset.

5.5 A Simple Decision Process for FS Method Selection

5.5.1 Introduction

Faced with a new dataset with many thousands of attributes, it would be of great help to a scientist to have good advice about what FS method to choose for that dataset. Usually, this decision could be made by referring to recent papers about feature selection, and finding methods that are popular and methods that seem to work well. But we can claim that this might miss important issues. Just because an FS method has worked well on some datasets, this does not mean it is the best FS method for the data in question. As we have seen, the DCV of the dataset could be seen as an important issue in this decision. In this part of the chapter we review the results that we have found from experiments, and see if this can be used to provide a tentative decision process. This process will only be relevant for the FS methods studied in the thesis. Also it may be only relevant for the datasets used in this thesis. But we can claim that this is likely to generalise to some extent. Of course, this way to produce a decision process can be repeated for other methods and datasets.

5.5.2 A Review of Performances on all Datasets

Table 5.12 records all the accuracies from all datasets used in the thesis. From this table we could easily find which algorithm is the best choice for which dataset. But for exploring deeper relationships between the DCV and the algorithm, we should rank the accuracies according to the performance of each algorithm and separate the datasets in a more manageable way.

General notes on following Table 5.12 to Table 5.21 are 1), “* *” highlight the best three algorithms for each data; 2), The lower average rank interprets the better performance, where rank 1 is best (for a dataset/row) and rank 8 is worst; 3), Bold average rank represents the best average rank overall algorithms (the best algorithm for this group of datasets).

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**	**CRFS Test**	**NFS Test**
0.280	78.709	90.150	90.206	90.262	90.306	90.273	89.994	92.131
0.378	85.264	89.945	88.788	89.705	89.805	89.511	89.966	89.805
0.381	80.412	91.697	89.627	93.044	89.594	93.122	92.521	93.122
0.388	86.722	89.182	89.538	89.560	89.683	89.649	88.914	89.215
0.451	88.837	90.206	90.762	90.161	89.872	90.295	90.462	89.816
0.512	89.438	90.095	90.306	90.495	90.929	90.262	90.006	90.395
0.513	94.224	92.51	90.918	93.311	90.774	93.734	94.28	93.678
0.560	91.497	91.241	91.664	91.931	91.263	92.354	91.875	92.31
0.567	99.065	99.154	93.155	99.032	92.498	98.965	99.221	93.745
0.663	92.577	93.767	91.107	91.875	91.152	93.378	93.467	93.089
0.602	51.333	47.334	44.0	41.333	60.667	44.666	42.0	39.333
0.828	83.53	70.588	87.059	71.765	69.412	66.471	73.529	65.294
0.882	95.168	89.128	71.812	85.906	92.483	84.027	91.811	85.503
0.099	79.36	79.2	69.12	78.080	80.16	78.24	69.12	78.4
0.185	46.904	51.190	45.952	49.047	50.476	53.571	50.238	56.904
0.335	63.2	64.16	64.16	63.36	64.0	64.16	62.4	63.36
0.349	89.76	88.480	83.04	90.720	89.12	88.96	87.52	91.68
0.399	86.24	91.039	84.16	91.2	89.76	92.0	90.72	90.24
0.449	86.24	92.640	82.72	89.760	91.04	91.679	85.6	88.64
0.496	90.24	89.120	87.52	93.76	91.04	91.36	90.72	88.96
0.51	91.523	70.728	64.635	71.1258	85.298	68.079	78.145	91.258
0.539	77.6	76.96	65.28	78.56	74.56	76.16	65.12	75.52
0.598	93.2	90.88	86.4	91.36	92.32	92.64	86.56	90.24
0.618	84.96	78.4	65.76	76.32	74.88	79.52	64.96	77.12
0.699	87.04	88.32	75.2	88.32	88.0	89.44	80.96	90.4
0.784	84.16	86.080	68.0	84.960	79.68	85.44	68.64	82.56
0896	98.88	93.12	83.68	91.84	88.96	93.28	81.0	92.16

Table 5.12. Summary of test set results for the main 8 FS and FB methods. This table records the test results (average over 5 test sets) for all the datasets used in the thesis. The purpose for building this table is to try to discovery the hidden information which could be used in a decision process.

Table 5.13 shows us the data from Table 5.12, but now it is expressed in terms of ranks, where “1” is best and “8” is worst for any particular dataset.

DCV	CFS Test	CFB Test	RFS Test	RFB Test	EAFS Test	EAFB Test	CRFS Test	NFS Test	Dataset
0.099	*2*	*3*	7	6	*1*	5	7	4	rOV1
0.185	7	*3*	8	6	4	*2*	5	*1*	PA
0.335	8	*3*	*1*	6	4	*2*	4	6	rOV2
0.349	*3*	6	8	*2*	4	5	7	*1*	rOV3
0.399	7	*3*	8	*2*	6	*1*	4	5	rOV4
0.449	6	*1*	8	4	*3*	*2*	7	5	rOV5
0.496	5	6	8	*1*	4	*2*	*3*	7	rOV6
0.451	8	4	*1*	5	6	*3*	*2*	7	OP1
0.388	8	6	4	*3*	*1*	*2*	7	5	OP3
0.381	8	5	6	*3*	7	*1*	4	*1*	OP5
0.378	8	*2*	7	5	*3*	6	*1*	*3*	OP9
0.280	8	6	5	4	*2*	*3*	7	*1*	OP8
0.51	*1*	6	8	5	*3*	7	4	*2*	IO
0.539	*2*	*3*	7	*1*	6	4	8	5	rOV7
0.598	*1*	5	8	4	3	*2*	7	6	rOV8
0.618	*1*	*3*	7	5	6	*2*	8	4	rOV9
0.699	6	*3*	8	*3*	5	*2*	7	*1*	rOV10
0.784	4	*1*	8	*3*	6	*2*	7	5	rOV11
0896	*1*	*3*	7	5	6	*2*	8	4	OV
0.663	5	*1*	8	6	7	*3*	*2*	4	OP7
0.567	*3*	*2*	7	4	8	5	*1*	6	OP0
0.560	6	8	5	*3*	7	*1*	4	*2*	OP2
0.513	*2*	6	7	5	8	*3*	*1*	4	OP4
0.512	8	6	4	*2*	*1*	5	7	*3*	OP6
0.602	*1*	4	8	5	*2*	7	*3*	6	CNS
0.828	*2*	5	*1*	4	6	7	*3*	8	AML/ALL
0.882	*2*	*3*	5	7	*1*	4	6	8	LUNG
Total	123	102	171	110	120	90	134	114	
Average	4.55	3.77	6.33	4.07	4.44	3.33	4.96	4.22	

Table 5.13. The information from Table 5.12, but in terms of ranks.

If we look at the bottom row of Table 5.13 we can see that the ranks of EAFB and CFB are the best overall. That means, if the correlation values and the size of dataset are unknown, the best two methods to consider are EAFB and CFB. However, the other methods generally perform well, except for RFS. But even RFS is the best ranked for some datasets. Overall, it is clearly better to consider the DCV of the dataset first before choosing the method, and also to consider the size (number of features) of the dataset.

To further investigate, we divide the datasets into several groupings, as follows:

1. By correlation: DCV less than 0.5;
2. By correlation: DCV more than 0.5;
3. By number of features: less than 1,000;
4. By number of features: more than 1,000;
5. DCV less than 0.5, less than 1,000 features;
6. DCV less than 0.5, more than 1,000 features;
7. DCV more than 0.5, less than 1,000 features;
8. DCV more than 0.5, more than 1,000 features.

Now we look at each group in turn.

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**	**CRFS Test**	**NFS Test**	**Dataset**
0.099	*2*	*3*	7	6	*1*	5	7	4	rOV1
0.185	7	*3*	8	6	4	*2*	5	*1*	PA
0.335	8	*3*	*1*	6	4	*2*	4	6	rOV2
0.349	*3*	6	8	*2*	4	5	7	*1*	rOV3
0.399	7	*3*	8	*2*	6	*1*	4	5	rOV4
0.449	6	*1*	8	4	*3*	*2*	7	5	rOV5
0.496	5	6	8	*1*	4	*2*	*3*	7	rOV6
0.451	8	4	*1*	5	6	*3*	*2*	7	OP1
0.388	8	6	4	*3*	*1*	*2*	7	5	OP3
0.381	8	5	6	*3*	7	*1*	4	*1*	OP5
0.378	8	*2*	7	5	*3*	6	*1*	*3*	OP9
0.280	8	6	5	4	*2*	*3*	7	*1*	OP8
Total	78	43	73	48	45	34	58	46	
Average	6.5	3.58	6.08	4.0	3.75	**2.83**	4.83	3.83	

Table 5.14. Twelve datasets with DCV < 0.5 that have been tested in our experiments. Some are from reduced ovarian datasets, and some from Optical digit datasets.

Table 5.14 shows the results on all datasets with DCV less than 0.5. It seems clear from this that EAFB is the best in this case. If we consider FS methods only (in the case that we are worried about the computation time of FB methods on very large datasets), then EAFS is best.

Table 5.15 shows the results on all datasets with DCV > 0.5. The results exhibited from this table, reveal the CFS is still the best solution while the correlation of dataset is above 0.5. EAFB has good performance on these datasets, but the first choice is clearly CFS.

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**	**CRFS Test**	**NFS Test**	**Dataset**
0.51	*1*	6	8	5	*3*	7	4	*2*	IO
0.539	*2*	*3*	7	*1*	6	4	8	5	rOV7
0.598	*1*	5	8	4	3	*2*	7	6	rOV8
0.618	*1*	*3*	7	5	6	*2*	8	4	rOV9
0.699	6	*3*	8	*3*	5	*2*	7	*1*	rOV10
0.784	4	*1*	8	*3*	6	*2*	7	5	rOV11
0896	*1*	*3*	7	5	6	*2*	8	4	OV
0.663	5	*1*	8	6	7	*3*	*2*	4	OP7
0.567	*3*	*2*	7	4	8	5	*1*	6	OP0
0.560	6	8	5	*3*	7	*1*	4	*2*	OP2
0.513	*2*	6	7	5	8	*3*	*1*	4	OP4
0.512	8	6	4	*2*	*1*	5	7	*3*	OP6
0.602	*1*	4	8	5	*2*	7	*3*	6	CNS
0.828	*2*	5	*1*	4	6	7	*3*	8	AML/ALL
0.882	*2*	*3*	5	7	*1*	4	6	8	LUNG
Total	45	59	98	62	75	56	76	68	
Average	**3.0**	3.93	6.53	4.13	5.0	3.73	5.06	4.53	

Table 5.15. Ranks of performance per dataset for each FS/FB strategy, represented by correlation value (DCV above 0.5).

When we look at datasets with < 1,000 features, in Table 5.16, there are three methods with close performance. It is interesting that the best method seems to be NFS. That is, if there are not many features, it is clearly best to make sure that all of them are used. If there are not many features, there seems to be no advantage to doing feature selection. It is also interesting that CFS seems to have the worst performance on these datasets. This is probably partly because the DCV of these datasets tends to be low.

Most of these datasets come from the optical digital data. The result for datasets less than 1,000 features is not very clear. Three algorithms (NFS, CRFS, EAFB) have similar good performance. Our suggestion for this kind of datasets is to apply three them if possible, and if only one method will be applied, it will be Null–feature – selection method. The majorities are above 0.5, but they do not rise very far, to a maximum of 0.663.

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**	**CRFS Test**	**NFS Test**	**Dataset**
0.663	5	*1*	8	6	7	*3*	*2*	4	OP7
0.567	*3*	*2*	7	4	8	5	*1*	6	OP0
0.560	6	8	5	*3*	7	*1*	4	*2*	OP2
0.513	*2*	6	7	5	8	*3*	*1*	4	OP4
0.512	8	6	4	*2*	*1*	5	7	*3*	OP6
0.51	*1*	6	8	5	*3*	7	4	*2*	IO
0.451	8	4	*1*	5	6	*3*	*2*	7	OP1
0.388	8	6	4	*3*	*1*	*2*	7	5	OP3
0.381	8	5	6	*3*	7	*1*	4	*1*	OP5
0.378	8	*2*	7	5	*3*	6	*1*	*3*	OP9
0.280	8	6	5	4	*2*	*3*	7	*1*	OP8
Total	65	52	62	45	53	39	40	38	
	5.90	4.72	5.63	4.09	4.81	3.54	3.63	3.45	

Table 5.16. Ranks of performance on the datasets less than 1,000 features.

Table 5.17 shows the performance on all datasets with 1,000 or more features. EAFB is the best in this case, with CFB second best.

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**	**CRFS Test**	**NFS Test**	**Dataset**
0.099	*2*	*3*	7	6	*1*	5	7	4	rOV1
0.185	7	*3*	8	6	4	*2*	5	*1*	PA
0.335	8	*3*	*1*	6	4	*2*	4	6	rOV2
0.349	*3*	6	8	*2*	4	5	7	*1*	rOV3
0.399	7	*3*	8	*2*	6	*1*	4	5	rOV4
0.449	6	*1*	8	4	*3*	*2*	7	5	rOV5
0.496	5	6	8	*1*	4	*2*	*3*	7	rOV6
0.539	*2*	*3*	7	*1*	6	4	8	5	rOV7
0.598	*1*	5	8	4	3	*2*	7	6	rOV8
0.618	*1*	*3*	7	5	6	*2*	8	4	rOV9
0.699	6	*3*	8	*3*	5	*2*	7	*1*	rOV10
0.784	4	*1*	8	*3*	6	*2*	7	5	rOV11
0896	*1*	*3*	7	5	6	*2*	8	4	OV
0.602	*1*	4	8	5	*2*	7	*3*	6	CNS
0.828	*2*	5	*1*	4	6	7	*3*	8	AML/All
0.882	*2*	*3*	5	7	*1*	4	6	8	LUNG
Total	58	55	107	64	67	51	94	76	
Average	3.62	3.43	6.68	4.0	4.18	**3.18**	5.87	4.75	

Table 5.17. Ranks of performance on the datasets no less than 1,000 features.

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**	**CRFS Test**	**NFS Test**	**Dataset**
0.451	8	4	*1*	5	6	*3*	*2*	7	OP1
0.388	8	6	4	*3*	*1*	*2*	7	5	OP3
0.381	8	5	6	*3*	7	*1*	4	*1*	OP5
0.378	8	*2*	7	5	*3*	6	*1*	*3*	OP9
0.280	8	6	5	4	*2*	*3*	7	*1*	OP8
Total	40	23	23	20	19	15	21	17	
Average	8.0	4.6	4.6	4.0	3.8	**3.0**	4.2	3.4	

Table 5.18. Ranks for datasets with DCV < 0.5 and with <1000 features. EAFB obtains the lowest average rank 3, and CFS performs very badly as the worst algorithm in each case.

When we look at datasets with DCV < 0.5 and with < 1,000 features in Table 5.18, there are some clear findings. Although, these must be taken to be quite tentative because the number of datasets is small and they are of the same general type. The finding is that EAFB is clearly the best method, and NFS is also very good. We saw similar results for table 5.16 (all datasets with < 1,000 features), but with DCV < 0.5 we see the poor performance of CFS more emphasised.

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**	**CRFS Test**	**NFS Test**	**Dataset**
0.663	5	*1*	8	6	7	*3*	*2*	4	OP7
0.567	*3*	*2*	7	4	8	5	*1*	6	OP0
0.560	6	8	5	*3*	7	*1*	4	*2*	OP2
0.513	*2*	6	7	5	8	*3*	*1*	4	OP4
0.512	8	6	4	*2*	*1*	5	7	*3*	OP6
0.51	*1*	6	8	5	*3*	7	4	*2*	IO
Total	25	29	39	25	34	24	19	21	
Average	4.16	4.83	6.5	4.16	5.66	4.0	**3.16**	3.5	

Table 5.19. Ranks for datasets with DCV > 0.5 and with <1000 features. The best algorithm is now CRFS and 2nd best is NFS. Performance of EAFS is quite poor, but EAFB is quite good.

For datasets with < 1,000 features but with DCV > 0.5 (Table 5.19), the results are interesting and different, with CRFS now seeming to be the best algorithm. But again, this is a small collection of datasets.

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**	**CRFS Test**	**NFS Test**	**Dataset**
0.099	*2*	*3*	7	6	*1*	5	7	4	rOV1
0.185	7	*3*	8	6	4	*2*	5	*1*	PA
0.335	8	*3*	*1*	6	4	*2*	4	6	rOV2
0.349	*3*	6	8	*2*	4	5	7	*1*	rOV3
0.399	7	*3*	8	*2*	6	*1*	4	5	rOV4
0.449	6	*1*	8	4	*3*	*2*	7	5	rOV5
0.496	5	6	8	*1*	4	*2*	*3*	7	rOV6
Total	38	25	48	27	26	19	37	29	
Average	5.42	3.57	6.85	3.85	3.71	**2.71**	5.28	4.14	

Table 5.20. Ranks for datasets with DCV < 0.5 and with > 1000 features. EAFB again exhibits strong ability for dealing with DCV less than 0.5, however many features.

Table 5.20 shows the situation when we have DCV < 0.5 and > 1,000 features. EAFB shows strong performance and CFS shows weak performance. The second best method is CFB and the third is EAFS. EAFS is therefore the best of the pure FS methods in this situation.

Finally, Table 5.21 shows us the results for the datasets with DCV > 0.5 and with > 1000 features. The strong performance of CFS in this case is obvious from this table.

DCV	**CFS Test**	**CFB Test**	**RFS Test**	**RFB Test**	**EAFS Test**	**EAFB Test**	**CRFS Test**	**NFS Test**	**Dataset**
0.539	*2*	*3*	7	*1*	6	4	8	5	rOV7
0.598	*1*	5	8	4	3	*2*	7	6	rOV8
0.618	*1*	*3*	7	5	6	*2*	8	4	rOV9
0.699	6	*3*	8	*3*	5	*2*	7	*1*	rOV10
0.784	4	*1*	8	*3*	6	*2*	7	5	rOV11
0896	*1*	*3*	7	5	6	*2*	8	4	OV
0.602	*1*	4	8	5	*2*	7	*3*	6	CNS
0.828	*2*	5	*1*	4	6	7	*3*	8	AML/All
0.882	*2*	*3*	5	7	*1*	4	6	8	LUNG
Total	20	30	59	37	41	32	57	47	
Average	**2.22**	3.33	6.55	4.11	4.55	3.55	6.33	5.22	

Table 5.21. Ranks for datasets with DCV > 0.5 and with > 1000 features. CFS has clearly the strongest performance in this case.

As shown in the Tables 5.14 to 5.21, the algorithm EAFB obtains the lowest overall rank, with performance usually among the top 3 for any dataset. Among the seven tables, the EAFB was the best performing method four times, and its worst performance was third, in two cases. EAFB is certainly a promising practical feature reduction method for common machine learning algorithms.

5.5.3 Summary

Table 5.22 gives an overall summary of the findings of the previous section. Figure 5.3 shows a graphical view of Table 5.22.

	Group of datasets	CFS	CFB	RFS	RFB	EAFS	EAFB	CRFS	NFS
1	ALL	4.55	3.77	6.33	4.07	4.44	**3.33**	4.96	4.22
2	DCV < 0.5	6.5	3.58	6.08	4.0	3.75	**2.83**	4.83	3.83
3	DCV > 0.5	**3.0**	3.93	6.53	4.13	5.0	3.73	5.06	4.53
4	Features less 1000	5.9	4.72	5.63	4.09	4.81	3.54	3.63	**3.45**
5	Features more 1000	3.62	3.43	6.68	4.0	4.18	**3.18**	5.87	4.75
6	DCV < 0.5 & Features < 1000	8	4.6	4.6	4.0	3.8	**3.0**	4.2	3.4
7	DCV > 0.5 & Features < 1000	4.16	4.83	6.5	4.16	5.66	4.0	**3.16**	3.5
8	DCV <0.5 & Features > 1000	5.42	3.57	6.85	3.85	3.71	**2.71**	5.28	4.14
9	DCV >0.5 & Features > 1000	**2.22**	3.33	6.55	4.11	4.55	3.55	6.33	5.22

Table 5.22. Nine groups of datasets are shown to divide categories of performance. The bold entry in each row shows the best algorithm for the type of data shown in that row.

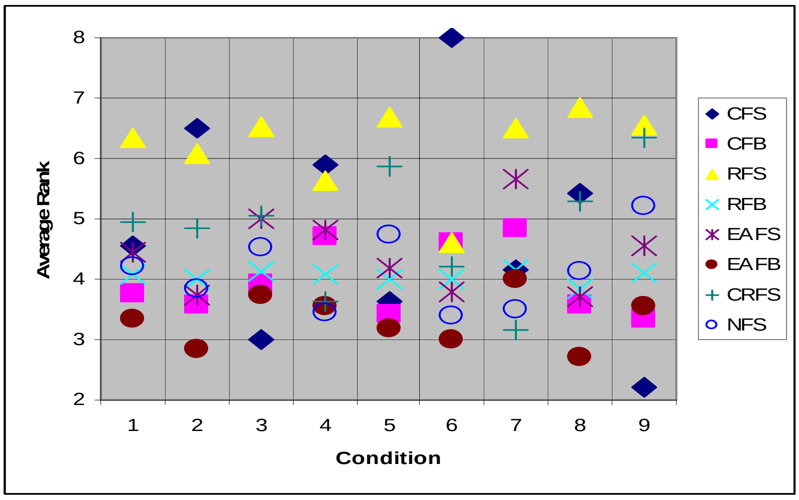


Figure 5.3. A graphical view of the data in Table 5.22. The different conditions on the *x* axis are the condition numbers in the left hand column of Table 5.22. The vertical distances give an idea of how clear the choice is for each condition. For example, in condition 9, the lowest (best) point is for CFS, showing that CFS is a clear best choice when DCV > 0.5 and features > 1000. The brown circle for EAFB is generally in a good position for all of the conditions.

In the conditions 1 (all datasets) and 2 (DCV < 0.5), EAFB seems the best algorithm on the datasets tested in this thesis. FB strategies overall tend to do well, as we can see in these results. In condition 3, the result reveals the CFS is still the best solution while the correlation of dataset is above 0.5. Beyond that, EAFB also has good performance on those datasets. The result for datasets with less than 1,000 features (condition 4) is not very clear, three algorithms (EAFB, CRFS, NFS) have similar performance. Our tentative suggestion for this kind of dataset is to test all three of the suggest methods if possible. But if only one method must be applied, it can simply be NFS (which is, do not do feature selection). In condition 5, EAFB is again the best solution for part of datasets, but CFB is close behind. EAFB is the best solution for all the conditions with correlation below 0.5 (conditions 2, 6 and 8). The combined strategy, CRFS (which we expect could useful combine the information from CFS and RFS) is the best overall for condition 7. In condition 9, CFS is clearly the best method.

5.5.4 A Suggested Decision Process

The experiments above show EAFB to be competitive in many different conditions. Because EAFB makes use of all the features in the training data, it avoids the possible problems of removing from consideration features that might be good. At the same time, it gives bias towards features that were found to be useful according to a short EA run beforehand. But in some conditions EAFB is clearly not the best method. The sum of the observations above comes to a simple decision process that we can summarise by Figure 5.4.

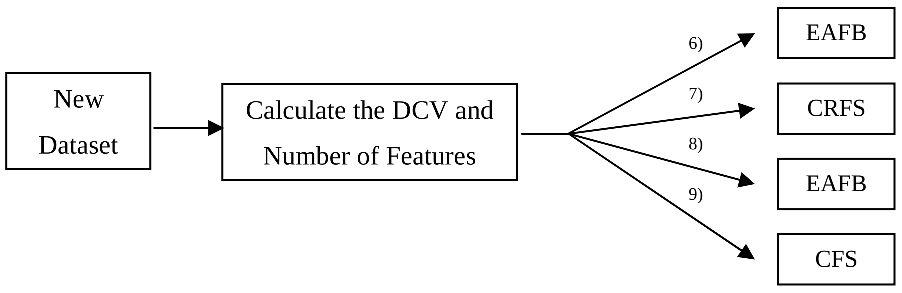


Figure 5.4. A basic decision process suggested by our findings. The condition numbers on the arrows refer to the condition numbers in Table 5.22.

This decision process is certainly not claimed to be generally true. Far more research needs to be done if a confident and accurate and useful decision process of this type can be considered for general use. But, based on the datasets we looked at and the FS and FB methods we studied here, the result is as provided in Figure 5.4. It can be easily argued that the validity of this decision process will extend to at least datasets that are similar to those studied here, for example, proteomics datasets of similar size. For such large proteomics datasets with low DCV (condition 8), we can feel quite confident that EAFB will be a much better choice than CFS.

5.6 Brief Discussion

The comparisons in this chapter were done over a collection of (mainly) many-attribute datasets, largely using proteomics data. The Feature bias strategies are very competitive compared to the traditional feature selection methods. EAFB seems to deserve more consideration than the other two FB strategies. One reason for this is the advantage of all FB methods, which is that they do not throw away any features. Another reason for this is that EAFB does not use a linear, simplified method to find bias values for features. The short EA run beforehand (from which features are chosen from the final population) is able (unlike, e.g., CFS) to exploit nonlinear relationships between subsets of features, through the standard evolutionary process. We can see that this might be especially useful when the DCV is low. That is, when the DCV is low, if there are relationships to be found between features then these relationships will not be obvious ones. They will not be the type of relationships that can be discovered by standard correlation methods.

We finished this chapter with the simple decision process that arises from our results. The reason for presenting this is mainly to present the following general idea, not the specific decision process itself. The idea is: the best choice of feature selection method for a dataset will depend on features of that dataset. To choose a good feature selection method, it is better to choose a method based on a process that understands the dataset, not based on, for example, publications that show that a FS is good, without considering the relationship between the dataset in that publication and the dataset in question. But, there is no guidance available, as we found in literature review, that considers how to choose the FS method based on statistical features of the dataset. In this thesis we show that statistical features of the dataset are important for this choice, and we show a simple decision process based on the datasets we studied and the algorithms we implemented. With more datasets and algorithms studied, a more refined simple decision process could be produced which will be a good guide for researchers who want to choose feature selection methods.

Chapter 6

Discussion and Future Work

6.1 Discussion

The large amount of data being increasingly produced, which may contain valuable hidden knowledge, continues to grow fast. Intuitively, the data stored in a database could be used to improve the decision-making process of an organization. Thus, there is an obvious demand for (semi-) automatic methods for extracting knowledge from data. This is the emergence of a now well-established field called data mining and knowledge discovery (Weiss S M and Indurkhya N (1998)). We focused on the data mining problem of *classification*. Bioinformatics data contains many very important classification problems, and it has been given increased attention recently for its complexity and size. Often, with the number of attributes so large in bioinformatics data, it is very hard to deal with for machine learning methods. However, the key information we could discover in bioinformatics data could help understand diseases better, and save the lives of patients. So, one task for computer science and machine learning is to find ways to find the relevant information among large datasets with many irrelevant features. Actually this includes the assumption that there are many irrelevant features, and that reduction of the number of attributes can be helpful at all. But this assumption is found to be true in many empirical studies so far.

Also, it is good if the discovered knowledge is comprehensible for the user. If the discovered "knowledge" is just a black box, which makes predictions without explaining them, the user may not trust it (Michie D, Spiegelhalter D J and Taylor C C (1994)). In the prediction task of bioinformatics, we hope the features which are explored in the data will help the biologists to design new models of gene expression or protein 3D structure. The major task of computer scientists is to find good features, however, the goal of biologist is to build a model to understand these features.

Thus we turn to feature selection methods, which are a key type of method for the general problem of dealing with many attribute datasets in general, not only bioinformatics. Feature selection methods based on basic statistics can find good features for many datasets, and this has been proven to very successful. The standard method is CFS (correlation-based feature selection), which directly discovers the features that are highly correlated with the target attribute in the dataset. High correlation means probably high performance when that set of features is used to represent the dataset in a machine learning process.

There are alternative methods. Relief F, for example, considers the distances between features. This strategy is good at looking for the most distinctive features among all. However, it is not always clear that distinctive features are useful for accurate classification.

Another strategy is to use an Evolutionary Algorithm (EA) for feature selection. In this method, we simply run the EA on the full feature set, and use the EA as the machine learning method. But, instead of using the evolved rules (or other structures) as the model to use for prediction, we just look at the features used in that model, in fact in all the models in the final population of the short EA run. In this way, indirectly, a number of features are 'selected' by the EA. The EA, over a number of iterations, has selected and preferred subsets of features that seem to work well together for the prediction task.

We tested these strategies on several datasets, and we tried to understand the results by looking at a simple statistical feature of each dataset: the dataset correlation value (DCV). We found that CFS was a good strategy, but only when the DCV was high. For datasets with low DCV, EAFS, and sometimes other methods, were better strategies. Considering these results, one clear thing is that the selection of features based on correlation is risky. The poor results for CFS on datasets with low DCV mean that (most likely) some features are being thrown away because they have low correlation with the target class, but these features can be important to get good accuracies. This leads to the idea that maybe it is good to never throw any features away, and this leads to the idea of using feature bias (FB) methods.

An FB method keeps all features selected, so all features are still available to the machine learning method. But, FB provides guidance to the machine learning method (it gives bias values) that can be used by the machine learning method to favour some features over others. Experiments showed that this approach performs very well.

We looked again at the results of all of the experiments, and tested the significance of two findings. One finding was that there seems to be clear correlation between the DCV and the performance of an FS algorithm (for any FS algorithm). Another finding was that, when the DCV is low, certain other algorithms are clearly better than CFS. We investigated the significance of these findings, and found that the statistics were generally in support of these findings, especially when we consider the real-world datasets. Finally, we derived a simple decision process from all of our results. This provides, based on looking at the DCV of a dataset and a basic consideration of the number of features, guidance about which FS method to choose, given the FS methods that we have studied. The decision process we derive is a set of very simple rules, and they only choose among the FS methods we have studied. The main point of this is the idea of it, to make a decision on FS based much on the DCV of the dataset. Future studies with more datasets and FS methods, also with maybe different statistical measures of the dataset, can produce refined versions of this decision process.

In the next sections, we summarise the thesis chapter by chapter, and then restate the contributions.

6.2 Overview of the Research

In chapter 1, we started by describing the motivations for this work, introducing the main themes upon which our research is inspired. It also introduced supervised learning and classification. Furthermore we introduced some basic concepts of bioinformatics and proteomics. The remaining chapters can be roughly divided into three parts: Chapter 2 is about a survey of algorithms and methods, mainly for feature selection. Chapters 3 and 4 show empirical studies, mainly on proteomics datasets, and showed basic results and findings. Then Chapter 5 analysed the significance of the results, and showed a first step towards how choosing feature selection method could be done by using the DCV of the dataset as a key factor in the choice.

In more detail, Chapter 2 surveyed feature selection techniques. We introduced feature selection algorithms as made of four parts: 1), starting subset of features; 2), selection methods for changing the subset of features; 3), evaluation strategy; 4), stop criterion. We then described three categories of FS method: 1), complete; 2), heuristic; 3), randomised, and two overall strategies for combining FS with machine learning (filter and wrapper). We gave an overview of the various existing state of the art techniques such as SVM-RFE and Instance-based algorithms, and discussed their strengths and weaknesses. Furthermore, we briefly described evolutionary algorithms, which later were applied as the machine learning method in out research.

Chapter 3 discussed the performance of five standard strategies on our collected datasets. The first technique was a statistical correlation-based approach, called CFS. We showed that only on highly-correlated data, this technique performed as good as, and often better than the other approaches. Among those five strategies, CFS (Relief-F, EAFS, CRFS, and NFS), EAFS appears the most successful of these techniques overall, since it was never worse than third ranked in any case (though NFS has similar overall performance). Also, we described some background experiments that informed the design of our EA for rule induction.

Chapter 4 outlined the rationale for a feature-bias (FB) approach, focused more on the opportunities of using features selected by the FS techniques, without discounting any features. FS is a common step in many classification and regression tasks. It is necessary because machine learning tools often cannot cope when the data has thousands of attributes. However, the strategy used by FS techniques is essentially binary – for any feature, the result of a prior FS stage is either “use this feature”, or “do not use this feature”. It is hoped that most “irrelevant” features are removed prior to the application of machine learning, and that the subsequent machine learning stage will be much faster (since there are fewer features to process) and also more successful (since many features will be removed by FS that seem unimportant for the classification task at hand). However, FS methods typically rely on standard statistical ideas and are from able to guarantee that all and only relevant features remain. FB, on the other hand, is an alternative approach in which we never entirely remove any feature from consideration. Instead, after a prior feature bias step, subsequent machine learning is guided by the bias values towards more preferably using some features rather than others. Chapter 4 showed promising results for this technique, even considering that we used a reduced number of iterations for the EA, to make up for the extra computation cost of including all features.

Chapter 5 looked at the results of chapters 3 and 4 again, to analyse if the main findings were significant. First we looked at the relationship between DCV and FS performance for different FS algorithms. On the reduced ovarian datasets, we did not find evidence for this relationship, but we noted that the reduced ovarian datasets have properties that may not be realistic for real world datasets. But on real datasets, we found that there was a significant correlation between the DCV and the performance of an FS algorithm. This was true for most FS methods. If this correlation was done for the MDCV, the results were not so significant, but there was still a tendency towards positive correlation. Then we tested a different but important finding. The results of our methods suggested that for datasets with low DCV, non-CFS methods were often definitely better than CFS. This means a negative correlation between: DCV and “the performance of method X relative to CFS”, where X is some specific other FS method. We found that this correlation was significant on the real datasets, especially for EAFS. Finally we looked at all of the results and derived a simple decision process, which might be useful when choosing between FS algorithms. The main point about this process is: it uses the DCV of a dataset to guide the choice of FS method.

6.3 Contributions

The scientific contributions of this study are as follows. The first contribution concerns the popular FS strategy: correlation-based feature selection (CFS). This strategy is often used, without question, to reduce the number of features from very large datasets. But we found, for example, among the datasets we studied, that it was never the best choice of FS method when the dataset had a low DCV. This means that using CFS in cases where a dataset has low DCV may do harm more than good. This is very important when we consider that sometimes the datasets involved are proteomics, or other bioinformatics data, and the task at had is to find good diagnostic tests or to better understand diseases. This contribution originates from chapter 3, it is confirmed in chapter 4, and we show in chapter 5 that it is supported by statistical significance.

A second contribution of the work is the description and application of feature bias (FB) techniques to the classification problems. In chapter 4, we defined several FB and combined strategies, and evaluated the performance of those strategies. We found that the FB version of a strategy was usually better than its basic FS version. CFB is good on high DCV datasets, for example, while CFS is poor. Meanwhile, EAFB seems to be the best strategy overall. The improvements of FB over FS come at a cost, however. Since a pure FB strategy still uses all of the features, it still has very high computational cost on very large datasets. But it might still be the best choice for medium sized datasets. Considering this, we also looked at combined FS and FB strategies, which trade off some of the cost for some of the benefit. We found that these generally performed well, but more research is needed. The many comparative experiments reported also comprise a contribution in this thesis.

Another contribution is the definition and use of the DCV, as a simple example of an indicator of a dataset that can be used to guide the choice of feature selection method. The studies of significance in chapter 5 added weight to this contribution. This chapter also explored a little the use of the MDCV as a variant on the DCV, but generally found that the DCV was more useful for prediction. But of course there might be other and better ways to find useful measures from the dataset.

Finally, we contribute a simple decision process for guiding the choice of FS method based on the DCV, and also the number of attributes. This decision process needs to be refined much, and it is only relevant for choosing between the FS and FB methods that we have tested. But there is no reason we cannot suggest it may be relevant for other many-attribute datasets. The main aspect of this contribution is the idea of it. Scientists needing to choose an FS strategy would benefit from a decision process that takes account of the DCV of the dataset, and further work could refine this decision process.

6.4 Future Work

Research on feature selection dates back to the 70s. This topic is not only a fundamental and traditional problem, but also a very challenging task despite more than a few decades' research efforts. Therefore, although the concept of feature selection is rather old, the application on proteomics data is quite new, mainly due to the recent advances in bioinformatics. Very often, feature selection methods are used by biologists to explore relationships inside their data. However, in this thesis we offer a new insight that views the relationship between the FS method and the dataset itself.

There are many limitations of our work which could be further tested in later work. The machine learning strategy we only test in our experiments is the evolutionary algorithm. It is easy to implement and was useful for our purposes of comparing FS methods. However, it could be claimed that our findings are only true when we are using this particular machine learning method. We would argue with this. The quality of accuracy that is possible for a machine learning method will depend mainly on the features from the dataset that it uses. However, it still might be important work to test some other widely-used machine learning strategies such as C4.5, SVM, and so on, to see the relationship between DCV and FS performance.

Another limitation is the limited number of large-scale datasets, especially with low DCV. Even though we have already tested 27 datasets in our experiments, the large-scale datasets with low DCV accounts for a very small proportion. Since low DCV datasets are the most interesting area – very challenging for CFS, and for any FS method, closer examination of a larger number of them will be important. This examination could first replicate our work, and then explore more thoroughly the performance of different FS, FB and variant algorithms on these datasets, and come up with a refined decision process for low DCV datasets.

Another are for future work is to investigate variants and better measures for a dataset. We have only tested mainly the DCV. This is the correlation value (the highest one) between one attribute and the target class. Obviously, datasets may be very different and still have the same DCV. E.g. in the case of a 10,000 attribute with DCV = 0.8, it could be that one attribute has correlation 0.8 with the target class, and all others are below 0.2. In another case it could be that they are all between 0.7 and 0.8. We might expect these two datasets to be suited to quite different FS methods. When we investigated this a little, by also testing MDCV, we found that the DCV was more useful. But this needs much more work. Research would be valuable to look at the distribution of correlation values in a dataset, and to somehow characterise the distribution (e.g. by mean and variance), and use this characterization of maybe two or more values as the guiding points for choosing FS strategy.

ABBREVIATIONS

CFB	Correlation-based feature bias method
CFS	Correlation-based feature selection method
CFS/CFB	The combination of correlation feature bias and feature selection method
CRFS	The combination strategy of CFS and RFS
DCV	Highest data correlation value
EAFB	Evolutionary algorithm feature bias method
EAFS	Evolutionary algorithm feature selection method
FB	Feature bias methods
FM	Feature management represents both feature selection and feature bias methods
FS	Feature selection methods
MDCV	Mean data correlation value
NFS	Null feature selection method
RFB	Relief feature bias method
RFB(3) / RFB(10)	The relief feature bias method with 3 nearest neighbour /10 nearest neighbour algorithm
RFS	Relief-F feature selection method
RFS/RFB	The combination of relief feature bias and feature selection method
RFS(3) / RFS(10)	The relief feature selection method with 3 nearest neighbour /10 nearest neighbour algorithm

GLOSSARY

Attribute (field, variable, feature)

A quantity describing an instance. An attribute has a domain defined by the attribute type, which denotes the values that can be taken by an attribute

Bioinformatics

The science of understanding and organising biological information by applying informatics techniques to data arising from biotechnology, usually on a large-scale

Classifier

A mapping from unlabeled instances to (discrete) classes. Classifiers have a form (e.g., decision tree) plus an interpretation procedure (including how to handle unknowns, etc.). Some classifiers also provide probability estimates (scores), which can be thresholded to yield a discrete class decision thereby taking into account a utility function

Data Mining

The term data mining sometimes refers to the whole process of knowledge discovery, and sometimes only to the specific machine learning phase

Feature

See Attribute

Machine Learning

In Knowledge Discovery, *machine learning* is most commonly used to mean the application of induction algorithms, which is one step in the knowledge discovery process. This is similar to the definition of empirical learning or inductive learning in Readings in Machine Learning by Shavlik and Dietterich T G. Note that in their definition, training examples are externally supplied, whereas here they are assumed to be supplied by a previous stage of the knowledge discovery process

Proteomics

Proteomics is the systematic large-scale analysis of protein expression under normal and perturbed states, and generally involves the separation, identification and characterization of all of the proteins in a cell or tissue sample

Rote Learning

A learning technique which avoids understanding of a subject and instead focuses on memorization. The major practice involved in rote learning is **learning by repetition**

Supervised Learning

Techniques used to learn the relationship between independent attributes and a designated dependent attribute (the label). Most induction algorithms fall into the supervised learning category

Systems Biology

This is the science interested in understanding the interactions and relationships between many parts of biological systems

REFERENCES

Aebersold R and Mann M (2003), *Mass Spectrometry-based Proteomics*. Nature, vol. 422, pp. 198–207.

Aha D W, Kibler D, and Albert M K (1991), *Instance Based Learning Algorithms*. Machine Learning, vol. 6, pp. 37–66.

Aha D W (1992), *Tolerating Noisy, Irrelevant and Novel Attributes in Instance-based Learning Algorithms*. International Journal of Man-Machine Studies, vol. 36, pp. 267–287.

Aha D W and Bankert R L (1995), *A Comparative Evaluation of Sequential Feature Selection Algorithms.* Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics, pp. 1–7.

Aha D W and Bankert R L (1996), *Improvement to a Neural Network Cloud Classifier.* Applied Meteorology, vol. 35, no. 11, pp. 2036–2039.

Almuallim H and Dietterich T G (1991), *Learning with Many Irrelevant Features*. 9th National Conference on Artificial Intelligence. MIT Press, pp. 547–552.

Anton H (2005), *Elementary Linear Algebra*. 9th Edition. Publisher: John Wiley and Sons.

Asuncion A and Newman D J (2008), *UCI Machine Learning Repository*. Irvine, University of California, School of Information and Computer Science. Retrieved from http://www.ics.uci.edu/ ~mlearn/MLRepository.html.

Baily T and Jain A K (1978), *A Note on Distance-weighted k-nearest neighbor Rules.* IEEE Transactions on Systems, Man& Cybernetics, vol. 8, no. 4, 311–313.

Bairoch A and Apweiler R (2000), *The SWISS-PROT Protein Sequence Database and Its Supplement TrEMBL*. Nucleic Acids Res, vol. 28, no. 1, pp. 45–80.

Barto A G and Sutton R S (1997), *Reinforce Learning in Artificial Intelligence*. Advances in Psychology, vol. 121, pp. 358–386.

Benson D A, Karsch–Mizrachi I, Lipman D J, Ostell J and Rapp B A (2000), *Wheeler DL.GenBank***.** Nucleic Acids Res, vol. 28, no. 1, pp. 8–15.

Beynon M, Chang C, Catalyurek U, Kurc T, Sussman A, Andrade H, Ferreira R and Saltz J (2002), *Processing Large-scale Multi-dimensional Data in Parallel and Distributed Environments*. Parallel Computing, vol. 28, no. 5, pp. 827–859.

Bins J and Draper B A (2001), *Feature Selection from Huge Feature Sets*. Proceedings of the Eighth International Conference on Computer Vision, vol. 2, pp. 159–165.

Blake C L and Merz CJ (1998), *UCI Repository of Machine Learning Databases*. Irvine, University of California, Department of Information and Computer Science. Retrieved from http://www.ics.uci.edu/ ~mlearn/MLRepository.html.

Boser B E, Guyon I and Vapnik V (1992), *A Training Algorithm for Optimal Margin Classifiers*. Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, ACM press, pp. 144–152.

Brazma F and Vilo J (2000), *Gene Expression Data Analysis***.** FEBS Letters, vol. 480, no. 1, pp. 17–24 A.

Brewer J K (1985), *Behavioural Statistics Textbooks: Source of Myths and Misconceptions?* Journal of Educational Statistics, vol. 10, pp. 252–268.

Bruggeman F J and Westerhoff H V (2007), *The Nature of Systems Biology*. Trends in Microbiology, Elsevier, vol. 15, pp. 45–50.

Bruha I and Franek F (1996), *Comparison of Various Routines for Unknown Attribute Value Processing: Covering Paradigm.* International Journal of Pattern Recognition and Artificial Intelligence, vol. 10, no. 8, pp. 939–955.

Burges C J C (1998), *A Tutorial on Support Vector Machines for Pattern Recognition*. Data Mining and Knowledge Discovery, vol. 2, no. 2, pp. 121–167.

Cagnoni S and Poli R (2006), *Genetic and Evolutionary Computation*. Intelligenza Artificiale, vol. 3, no. 1–2, pp. 94–101.

Celis J E and Bravo R (1984), *Two-Dimensional Gel Electrophoresis of Proteins: Methods and Applications*. New York, Academic Press.

Chen X (2003), *An Improved Branch and Bound Algorithm for Feature Selection*. Pattern Recognition Letters, vol. 24, pp. 1925–1933.

Chen Y W and Lin C J (2006), *Feature Extraction, Foundations and Applications.* Chapter Combining SVMs with various feature selection strategies, in Guyon I, Gunn S, Nikravesh M, Zadeh LA Eds, Springer–Verlag, Berlin.

Cortes C and Vapnik V (1995), *Support-Vector Networks*. Machine Learning vol. 20, no. 3, pp. 273–297.

Dash M and Liu H (1997), *Feature Selection for Classification*. Intelligent Data Analysis, vol. 1, no. 3, pp. 131–156.

Dash M and Liu H (2003), *Consistency-based Search in Feature Selection,* Artificial Intelligence, vol. 151, no. 1–2, pp. 155–176.

Devore J and Peck R (1994), *Introductory Statistics,* Second edition, West Publisher Co..

Dietterich T G (1997), *Machine Learning Research: Four current directions*. AI Magazine, vol. 18, no. 4, pp. 97–136.

Dorak M T (2006), *Common Concepts in Statistics*. Retrieved from http://dorakmt.tripod.com/mtd/glosstat.html.

Dougherty J, Kohavi R and Sahami M (1995), *Supervised and Unsupervised Discretisation of Continuous Features*. Machine Learning: Proceedings of the Twelfth International Conference, Morgan Kaufmann, pp. 194–202.

Duan K and Rajapakse J C (2005), *SVM-RFE Peak Selection for Cancer Classification with Mass Spectrometry Data*. APBC, pp. 191–200.

Elio R and Watanabe L (1991), *An Incremental Deductive Strategy for Controlling Constructive Induction in Learning from Examples.* Machine Learning, vol. 7, no. 1, pp. 7–44.

Fogel D B, Angeline P, Porto V, III, E W, and Boughton E (1999), *Using Evolutionary Computation to Learn About Detecting Breast Cancer*. Proceedings of the Congress of Evolutionary Computation (CEC–1999), in Angeline *et al*., Eds, vol. 3, pp. 1749–1754.

Fogel L J, Owens A J, and Walsh M J (1966), *Artificial Intelligence through Simulated Evolution*. John Wiley, New York.

Ferri F J, Pudil P, Hatef M, and Kittler J (1994), *Comparative Study of Techniques for Large–scale Feature Selection*. Pattern Recognition in Practice IV, Multiple Paradigms, Comparative Studies and Hybrid Syst., in E S Gelsema and L S Kanal, Eds. Amsterdam: Elsevier, pp. 403–413.

Fisher D H (1997), *Machine Learning*. Proceedings of the Fourteenth International Conference (ICML'97), Morgan Kaufmann, pp. 296–304.

Forman G (2003), *An Extensive Empirical Study of Feature Selection Metrics for Text Classification.* Journal of Machine Learning Research, vol. 3, pp. 1289–1305.

Freitas A A (2001), *A Survey of Evolutionary Algorithms for Data Mining and Knowledge Discovery*. Advances in Evolutionary Computation, in A. Ghosh and S.S. Tsutsui, Eds, New York, Springer–Verlag.

Gadat S and Younes L (2007), *A Stochastic Algorithm for Feature Selection in Pattern Recognition*. MIT Press Cambridge, MA, USA, pp. 509–547.

Gao S Y (2002), *S–TSKFNN: A Novel Self-organizing Fuzzy Neural Network Based on the TSK Fuzzy Rule Model*. School of Computer Engineering, Intelligent Systems, Laboratory, Nanyang Technological University, Singapore.

Goldberg DE (1989), *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison–Wesley, MA.

Golub T R, Slonim D K, Tamayo P, Huard C, Gaasenbeek M, Mesirov J P, Coller H, Loh M L, Downing J R, Caligiuri M A, BloomÞeld C D and Lander E S (1999), *Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring*. Science, vol. 286, pp. 531–537.

Gordon G J, Jensen R V, Hsiao L L, Gullans S R, Blumenstock J E, Ramaswamy S, Richards W G, Sugarbaker D J, Bueno R (2002), *Translation of Microarray Data into Clinically Relevant Cancer Diagnostic Tests Using Gene Expression Ratios in Lung Cancer and Mesothelioma*. Cancer Res, vol. 62, pp. 4963–4967.

Guyon I, Weston J, Barnhill S, and Vapnik V (2002), *Gene Selection for Cancer Classification Using Support Vector Machines*. Machine Learning, vol. 46, pp. 389–422.

Guyon I and Elisseeff A (2003), *An Introduction to Variable and Feature Selection*. Machine Learning, special issue on variable and feature selection, vol. 3, pp. 1157–1182.

Guyon I, Gunn S, Hur A B and Dror G (2004), *Result Analysis of the NIPS 2003 Feature Selection Challenge*. Proceedings of the Neural Information Processing Systems, pp. 545–552.

Guyon I**, Aliferis C and Elissee A (2007),** *Causal Feature Selection*. Chapman and Hall/CRC Press, pp. 63–82.

Hall M A (1998), *Correlation-based Feature Selection Machine Learning*. Ph.D Thesis, Department of Computer Science, University of Waikato, Hamilton, New Zealand.

Hall M A (2000), *Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning.* Proceeding of 17th International Conference on Machine Learning, pp. 359–366.

Hingorani S, Petricoin III E, Maitra A, Rajapakse V, King C, Jacobetz M, Ross S, Conrads T, Veenstra T and Hitt B (2003), *Preinvasive and Invasive Ductal Pancreatic Cancer and Its Early Detection in the Mouse*. Cancer Cell, vol. 4, no. 6, pp. 437–450.

Holland J H (1975), *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI.

Huerta M, *et al.* (2000), *NIH Working Definition of Bioinformatics and Computational Biology*. The Biomedical Information Science and Technology Initiative Consortium (BISTIC), Definition Committee of National Institutes of Health (NIH).

Ideker T, *et al.* (2000), *Testing for Differentially-expressed Genes by Maximum Likelihood Analysis of Microarray Data.* Computer Bioinformatics, vol. 7, pp. 805–817.

Ideker T, *et al.* (2001), *Integrated Genomic and Proteomic Analysis of a Systematically Perturbed Metabolic Network*. Science, vol. 292, pp. 929–934.

Ishibuchi H, Nakashima T and Murata T (1999), *Performance Evaluation of Fuzzy Classifier Systems for Multidimensional Pattern Classification Problems*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 29, no. 5, pp. 601–618.

Jain A and Zongker D (1997), *Feature Selection: Evaluation, Application and Small Sample Performance*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 2, pp. 153–158.

John G H, Kohavi R and Pfleger K (1994), *Irrelevant Features and the Subset Selection Problem.* International Conference on Machine Learning, pp. 121–129.

Jones S S and Smith L B (1991), *Object Properties and Knowledge in Early Lexical Learning*. Child Development, vol. 62, pp. 499–516.

Jong K, *et al.* (2004), *Feature Selection in Proteomic Pattern Data with Support Vector Machines*. Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, pp. 41–48.

Juliusdottir T, Keedwell E, Corne D W and Narayanan A (2005), *Two-Phase EA/k-NN for Feature Selection and Classification in Cancer Microarray Datasets Computational Intelligence in Bioinformatics and Computational Biology*. CIBCB ‘05. Proceedings of the 2005 IEEE Symposium on Publication, ISBN: 0–7803–9387–2.

Kaynak C (1995), *Methods of Combining Multiple Classifiers and Their Applications to Handwritten Digit Recognition*, MSc Thesis, Institute of Graduate Studies in Science and Engineering, Bogazici University.

Khayat O, Shahdoosti H R and Khosravi M H, (2008*),* *Stable Relief in Feature Weighting*. 7th WSEAS International Conference on artificial intelligence, knowledge engineering and data bases (AIKED’08), University of Cambridge, UK, pp. 193–197.

Kira K and Rendell L A (1992a), *The Feature Selection Problem: Traditional Methods and New Algorithm*. Proceedings of AAAI’92, pp. 129–134.

Kira K and Rendell L A (1992b), *A Practical Approach to Feature Selection*. In D. Sleeman, and P. Edwards (Eds.), Machine Learning: Proceedings of International Conference (ICML’92)), Morgan Kaufmann, pp. 249–256.

Kitano H (2002), *Systems Biology: A Brief Review.* Science vol. 295, pp. 1662–1664.

Koller D and Sahami M (1997), *Toward Optimal Feature Selection.* Proceeding Of 13th International Conference on Machine Learning, Morgan Kaufmann, pp. 284–292.

Kononenko I (1994), *Estimating Attributes: Analysis and Extensions of RELIEF.* Machine Learning: ECML’94, Springer LNCS, vol. 784, pp. 156–160.

Kononenko I, Simec E, and Robnik S M (1997), *Overcoming the Myopia of Inductive Learning Algorithms with RELIEF-F.* Applied Intelligence, vol. 7, pp. 39–55.

Koza and John R (1992), *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, MA: MIT Press.

Kohavi R, John G H, Long R, Manley D and Pfleger K (1994), *A Machine Learning Library in C++*. Tools with Artificial Intelligence. IEEE Computer Society Press, pp. 740–743.

Kohavi R (1995a), *The Power of Decision Tables*. European Conference on Machine Learning, 1995, vol. 914, pp. 174–189.

Kohavi R (1995b), *Wrappers for Performance Enhancement and Oblivious Decision Graphs*. PhD thesis, Stanford University.

Kohavi R and John G H (1996), *Wrappers for Feature Subset Selection*. Artificial Intelligence, special issue on relevance, vol. 97, no. 1–2, pp. 273–324.

Kohavi R, Langley P, and Yun Y (1997a), *The Utility of Feature Weighting in Nearest-neighbor Algorithms*. Proceedings of the Ninth European Conference on Machine Learning, Prague, Springer–Verlag.

Kohavi R, John G H (1997b), *Wrappers for Feature Subset Selection*. Artificial Intelligence, vol. 97, no. 1–2, pp. 273–324.

Kudo M and Sklansky J (2000), *Comparison of Algorithms that Select Features for Pattern Classifiers*. Pattern Recognition, vol. 33, pp. 25– 41.

Lamb G S (1984), *What You Always Wanted to Know about Six but were Afraid to Ask*. The Journal of Irreproducible results, vol. 29, pp. 18–20.

Langley P (1994), *Selection of Relevant Features in Machine Learning*. Proceedings of the AAAI Fall Symposium on Relevance, AAAI Press, pp. 140–144.

Langley P and Sage S (1997), *Scaling to Domains with Irrelevant Features*. In R. Greiner, Ed, Computational Learning Theory and Natural Learning Systems, MIT Press, vol. 4.

Leardi R and Gonzales A L (1998), *Genetic Algorithms Applied to Feature Selection in PLS Regression: How and When to Use Them.* Chemometrics and Intelligent Laboratory Systems, vol. 41, pp. 195–207.

Li X (2007), *An Analysis Of Chinese EFL Learners' Beliefs About The Role Of Rote Learning.* Vocabulary Learning Strategies. Retrieved from http://www.asian–efl–journal.com/xiuping_11–05_thesis.pdf.

Liu H and Setiono R (1996), *A Probabilistic Approach to Feature Selection: A Filter Solution*. Proceeding of 13th International Conference on Machine Learning, Bari, Italy, pp. 319–327.

Liu H and Motoda H (1998), *Feature Selection for Knowledge Discovery and Data Mining*. Boston: Kluwer Academic Publishers.

Li T, *et al.* (2004), *A Comparative Study of Feature Selection and Multiclass Classification Methods for Tissue Classification Based on Gene Expression*. Bioinformatics, vol. 20, pp. 2429–2437.

Lomax, R. G. (2001), *An Introduction to Statistical Concepts for Educational and Behavioural Sciences*, Mahwah, NJ, Lawrence Erlbaum Associates.

Luo S and Corne D W (2008), *Feature Selection Strategies for Poorly Correlated Data: Correlation Coefficient Considered Harmful.* In Kazovsky *et al.* (eds.), Proceedings of the 7th WSEAS Int'l Conf. on Artificial Intelligence, Knowledge Engineering and Databases (AIKED'08), Cambridge, UK, WSEAS Press, ISBN: 978–960–6766–41–1, ISSN: 1790–5109, pp. 226—231.

Luscombe N M, Greenbaum D and Gerstein M (2001), *What is Bioinformatics? A Proposed Definition and Overview of the Field.* Methods Inf. Med, vol. 40, pp. 346–358.

Lyerly (1952), *The Average Spearman Rank Correlation Coefficient.* Psychometrika, vol. 17, no. 4, pp. 421–428.

McCallum A and Nigam K (1998). *A Comparison of Event Models for Naive Bayes Text Classification.* AAAI–98 Workshop on Learning for Text Categorization.

MacFarland T W (1998), *Spearman's Rank-Difference Coefficient of Correlation.* Retrieved from http://www.nyx.net/~tmacfarl/STAT_TUT/spearman.ssi.

Mamdani E H (1974), *Application of Fuzzy Algorithms for Control of Simple Dynamic Plant*. Proceeding of IEE, vol. 121, pp. 1585–1587.

Mao Y, Zhou X, Pi D, Sun Y and Wong T C (2005), *Automated Recognition of Cellular Phenotypes by Support Vector Machines with Feature Reduction,* Journal of Biomedicine and Biotechnology, vol. 2, pp. 160–171.

Michalski R S, Anderson J R, Carbonell J G and Mitchell T M (1983), *Machine Learning: An Artificial Intelligence Approach.* Morgan Kaufmann.

Michalski R S and Kodratoff Y (1990), *Machine Learning: An Artificial Intelligence Approach,* Morgan Kaufmann, San Francisco, vol.3.

Michie D, Spiegelhalter D J and Taylor C C (1994), *Machine Learning, Neural and Statistical Classification*. New York, Ellis Horwood.

Miller A J (1990), *Subset Selection in Regression*. Chapman and Hall, New York.

Mitchell T (1997), *Machine Learning*. Mcgraw–Hill.

Molina L, Belanche L and Nebot A (2002), *Feature Selection Algorithms: A Survey and Experimental Evaluation.* Proceeding of IEEE International Conference on Data Mining, IEEE, pp. 306–313.

Newell A, and Simon H A (1956), *The Logic Theory Machine*. IRE Transactions on Information Theory, vol. 2, no. 3, pp. 61–79.

Nilsson N J (1996), *Introduction to Machine Learning.* Robotics Laboratory Department of Computer Science, Stanford University, Stanford, CA, December, 1996. Retrieved from http://ai.stanford.edu/people/nilsson/mlbook.html.

Pandey A and Mann M (2000), *Proteomics to Study Genes and Genomes*. Nature, vol. 405, pp. 837–846.

Petricoin E, Ardekani A, Hitt B, Levine P, Fusaro V, Steinberg S, Mills G, Simone C, Fishman D and Kohn E (2002), *Use of Proteomic Patterns in Serum to Identify Ovarian Cancer*. The Lancet, vol. 359, no. 9306, pp. 572–577.

Pomeroy S L, ***et al*****. (2002),** *Prediction of Central Nervous System Embryonal Tumour Outcome Based on Gene Expression II*. Nature, vol. 415, pp. 436–442.

Pompe U and Kononenko I (1995), *Linear Space Induction in First Order Logic with Relief-F*. In G. Della Riccia, R. Kruse, and R. Viertl, Eds. Mathematical and Statistical Methods in Artificial Intelligence, CISM Courses and Lectures, Springer-Verlag.

Pudil P and Novovicova J (1998), *Novel Methods for Subset Selection with Respect to Problem Knowledge*. IEEE Intelligent Systems, pp. 66–74.

Quinlan J R (1983), *Learning Efficient Classification Procedures and their Application to Chess End Games*. Machine Learning: An Artificial Intelligence Approach, pp. 463–482.

Quinlan J R (1986), *Induction of Decision Trees*. Machine Learning, vol. 1, pp. 81–106.

Quinlan J R (1989), *Unknown Attribute Values in ID3*. International Conference ML, Morgan Kaufmann**,** pp. 164–168.

Quinlan J R (1993), *C4.5: Programs for Machine Learning*. Morgan Kaufmann, Los Altos, California, 1993.

Rechenberg I (1973), *Evolution Strategies: Optimierung Technischer Systeme Nach Prinzipien Der Biologischen Evolution*, Frommann–Holzboog, Stuttgart.

Reichhardt T (1999), *It is Sink or Swim as a Tidal Wave of Data Approaches*. Nature, vol. 399, no. 6736, pp. 17–20.

Rich E and Knight K (1991), *Artificial Intelligence*. McGraw–Hill, 1991.

Robnik S M and Kononenko I (1997), *An Adaptation of Relief for Attribute Estimation in Regression*. Proceedings of the Fourteenth International Conference on Machine Learning, Morgan Kaufmann, pp. 296–304.

Saeys Y (2004), *Feature Selection for Classification of Nucleic Acid sequences*, PhD thesis.

Scherf M and Brauer W (1997), *Feature Selection by Means of a Feature Weighting Approach*. Technical Report FKI–221–97, Technische Universität München, Munich.

Schwefel H P (1981), *Numerical Optimization of Computer Models*. John Wiley, Chichester, U.K.

Schwefel H P (1995), *Evolution and Optimum Seeking*. John Wiley.

Sigillito V G, Wing S P, Hutton L V and Baker K B (1989), *Classification of Radar Returns from the Ionosphere Using Neural Networks*. Johns Hopkins APL Technical Digest, vol. 10, pp. 262–269.

Sikonja M R and Kononenko I (2003), *Theoretical and Empirical Analysis of Relief-F and RRelief-F*. Machine Learning, vol. 53, pp. 23–69.

Singhi S K and Liu H (2006), *Feature Subset Selection Bias for Classification Learning.* Proceedings of the 23rd International Conference on Machine learning, Pittsburgh, Pennsylvania, pp. 849–856.

Skalak's D B (1994), *Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithms.* In: W.W. Cohen and H. Hirsh, Eds, Proceedings of the Eleventh International Conference, pp. 293–301.

Spears W A, De Jong K A, Baeck T, Fogel D B, and De Garis H (1993), *An Overview of Evolutionary Computation.* Presented at European Conference on Machine Learning, P. V. Brazdil Eds, Springer Verlag, vol. 667, pp. 442–459.

Sun Y and Li J (2006), *Iterative RELIEF for Feature Weighting*. Proceedings of 23rd International Conference Machine Learning, pp. 913–920.

Takagi T and Sugcno M (1985), *Fuzzy Identification of Systems and its Applications to Modelling and Control.* IEEE Trans. on Systems, Man& Cybernetics. vol. 15, pp. 116–132.

Tang Y, Zhang Y, Huang Z and Hu X (2005), *Atlanta Granular SVM–RFE Gene Selection Algorithm for Reliable Prostate Cancer Classification on Microarray Expression Data*. From Georgia State University, BIBE, pp. 290–293.

Ungsik Yu, Sung Hoon Lee, Young Joo Kim and Sangsoo Kim (2004). *Bioinformatics in the Post-genome Era*. Journal of Biochemistry and Molecular Biology, vol. 37, no. 1, pp. 75–82.

Ursem R K (2003), *Models for Evolutionary Algorithms and their Applications in System Identification and Control Optimization*. PhD thesis, University of Aarhus, EVALife, Dept. of Computer Science, Denmark.

Vapnik V (1995), *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Weiss S M and Indurkhya N (1998), *Predictive Data Mining: a Practical Guide*. Morgan Kaufmann.

Weston J, Mukherjee S, Chapelle O, Pontil M, Poggio T, and Vapnik V (2001), *Feature Selection for SVMs*. Advances in Neural Information Processing Systems, Morgan Kaufmann, vol. 13.

Wettschereck D, Aha D W and Mohri T (1997), *A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms*. Artificial Intelligence Review, vol. 11, pp. 273–314.

Wu Y and Zhang A (2004*)*, *Feature Selection for Classifying High-Dimensional Numerical Data*. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04), pp. 251–258.

Xing E P, Jordan M and Karp R M (2001), *Feature Selection for High-dimensional Genomic Microarray Data*. Proceedings of the Eighteenth International Conference on Machine Learning, pp. 601–608.

Xing E P and Karp R M (2001), *CLIFF: Clustering of High-dimensional Microarray Data via Iterative Feature Filtering Using Normalised Cuts*. Proceedings of the Nineteenth International Conference on Intelligent Systems for Molecular Biology, vol. 17, pp. 306–315.

Yang J and Honavar V (1998), *Feature Subset Selection Using a Genetic Algorithm*. IEEE Intelligent Systems, vol. 13, pp. 44–49.

Yang Y and Pedersen J O (1997), *A Comparative Study on Feature Selection: Text Categorization.* Proceedings of the Fourteenth International Conference on Machine Learning, pp. 412–420.

Yu L and Liu H (2003), *Feature Selection for High-dimensional Data: a Fast Correlation-based Filter Solution*. Proceedings of the twentieth International Conference on Machine Learning, pp. 856–863.

Yu L, and Liu, H (2004*),* *Efficient Feature Selection via Analysis of Relevance and Redundancy*. J Machine Learning Res, vol. 5, pp. 1205–1224.

Zadeh L A (1994), *Soft Computing and Fuzzy Logic*. IEEE Software, pp. 48–56.

Zeng F, Yap R and Wong L (2002), *Using Feature Generation and Feature Selection for Accurate Prediction of Translation Initiation Sites*. Genome Informatics, vol. 13, pp. 192–200.