

# **FRACTAL DIMENSION FOR CLUSTERING AND UNSUPERVISED AND SUPERVISED FEATURE SELECTION**

A thesis submitted to Cardiff University  
in candidature for the degree of

**Doctor of Philosophy**

by

**Moises Noe Sanchez Garcia**

Manufacturing Engineering Centre  
School of Engineering  
Cardiff University  
United Kingdom

March 2011

UMI Number: U585577

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI U585577

Published by ProQuest LLC 2013. Copyright in the Dissertation held by the Author.  
Microform Edition © ProQuest LLC.

All rights reserved. This work is protected against  
unauthorized copying under Title 17, United States Code.



ProQuest LLC  
789 East Eisenhower Parkway  
P.O. Box 1346  
Ann Arbor, MI 48106-1346

# ABSTRACT

Data mining refers to the automation of data analysis to extract patterns from large amounts of data. A major breakthrough in modelling natural patterns is the recognition that nature is fractal, not Euclidean. Fractals are capable of modelling self-similarity, infinite details, infinite length and the absence of smoothness.

This research was aimed at simplifying the discovery and detection of groups in data using fractal dimension. These data mining tasks were addressed efficiently. The first task defines groups of instances (clustering), the second selects useful features from non-defined (unsupervised) groups of instances and the third selects useful features from pre-defined (supervised) groups of instances. Improvements are shown on two data mining classification models: hierarchical clustering and Artificial Neural Networks (ANN).

For clustering tasks, a new two-phase clustering algorithm based on the Fractal Dimension (FD), compactness and closeness of clusters is presented. The proposed method, uses self-similarity properties of the data, first divides the data into sufficiently large sub-clusters with high compactness. In the second stage, the algorithm merges the sub-clusters that are close to each other and have similar complexity. The final clusters are obtained through a very natural and fully deterministic way.

The selection of different feature subspaces leads to different cluster interpretations. An unsupervised embedded feature selection algorithm, able to detect relevant and redundant features, is presented. This algorithm is based on the concept of fractal dimension. The level of relevance in the features is quantified using a new proposed entropy measure, which is less complex than the current state-of-the-art technology. The proposed algorithm is able to maintain and in some cases improve the quality of the clusters in reduced feature spaces.

For supervised feature selection, for classification purposes, a new algorithm is proposed that maximises the relevance and minimises the redundancy of the features simultaneously. This algorithm makes use of the FD and the Mutual Information (MI) techniques, and combines them to create a new measure of feature usefulness and to produce a simpler and non-heuristic algorithm. The similar nature of the two techniques, FD and MI, makes the proposed algorithm more suitable for a straightforward global analysis of the data.

*to my mum.*

## ACKNOWLEDGEMENTS

I would like to express my deepest and sincere gratitude to my supervisor, Prof. D. T. Pham, for giving me guidance and support during all the stages of my research.

Special thanks to Dr. Michael S. Packianather and Dr. Marco Castellani with whom I always was able to find fruitful discussions regarding my work.

I would like also to thank my family for their support and encouragement through all my life and my girlfriend Rach for all the love.

Thanks to the members of the MEC machine learning group for sharing interesting ideas and useful comments in our weekly meetings.

Thanks to the colleagues in the lab I worked for helping to keep my spirit up.

I would like to thank CONACYT (CONsejo NAcional de Ciencia Y Tecnologia) Mexico for giving me the financial means of pursuing a Ph.D education.

# DECLARATION

## DECLARATION

This work has not previously been accepted in substance for any degree and is not concurrently submitted in candidature for any degree.

Signed .....  ..... (Moises Noe Sanchez Garcia) Date 31/05/2011

## STATEMENT 1

This thesis is being submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy (PhD).

Signed .....  ..... (Moises Noe Sanchez Garcia) Date 31/05/2011

## STATEMENT 2

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

Signed .....  ..... (Moises Noe Sanchez Garcia) Date 31/05/2011

## STATEMENT 3

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

Signed .....  ..... (Moises Noe Sanchez Garcia) Date 31/05/2011

# TABLE OF CONTENTS

	<b>Page</b>
<b>Abstract</b>	<b>i</b>
<b>Acknowledgements</b>	<b>iv</b>
<b>Declaration</b>	<b>v</b>
<b>List of Figures</b>	<b>ix</b>
<b>List of Tables</b>	<b>xii</b>
<b>Abbreviations</b>	<b>xiii</b>
<b>Nomenclature</b>	<b>xv</b>
<b>Chapter 1 Introduction</b>	<b>1</b>
1.1 Background	1
1.2 Fractal Mining	3
1.3 Statistical Entropy	3
1.4 Mutual Information	4
1.5 Motivation	4
1.6 Aim and Objectives	5
1.7 Outline of the thesis	6
<b>Chapter 2 Background</b>	<b>8</b>
2.1 Fractal Dimension	8
2.1.1 Self-Similarity	8
2.1.2 Box Counting Dimension	10
2.2 Clustering	11
2.2.1 Clustering techniques	12

2.2.2	Number of clusters	14
2.2.3	Cluster Validity	25
2.3	Feature Selection Approaches	15
2.3.1	Filter Approach	16
2.3.2	Wrapper Approach	17
2.3.3	Embedded approach	18
2.4	Unsupervised Feature Selection Approaches	20
2.5	Unsupervised Feature Relevance	21
2.6	Statistical Entropy	22
2.6.1	Entropy in Data Mining	23
2.6.2	Entropy as Relevance Measure	25
2.7	Search Techniques Overview	25
2.7.1	Sequential Selection Algorithms (SSA)	26
2.7.1	Second generation of SSA	27
2.8	Relevance in Features	31
2.8.1	Relevance Approaches in FS	23
2.8.2	Relevance versus Optimality	25
2.9	Feature Redundancy	37
2.10	Mutual Information	39
2.11	Redundancy Feature Analysis	37
2.10	Redundancy Feature Analysis	39
2.11	Summary	40
<b>Chapter 3</b>	<b>Divisive Fractal Clustering Approach</b>	<b>41</b>
3.1	Preliminaries	41
3.2	Hierarchical Clustering	42
3.2.1	Agglomerative analysis	45
3.2.2	DIANA (Divisive Analysis)	46



3.3 Fractal Clustering	39
3.4 Proposed Algorithm	48
3.4.1 Second Phase	50
3.4.2 Stopping Criterion	50
3.4.3 Number of clusters	50
3.4.4 Refining Step	52
3.4.5 Evaluation Measure	52
3.4.5 DIANA (Divisive Analysis)	46
3.5 Experimental Results	52
3.3 Summary	55
<b>Chapter 4 Unsupervised Feature Selections</b>	<b>56</b>
4.1 Preliminaries	56
4.2 Unsupervised Feature Selections	57
4.3 Unsupervised Relevance and Redundancy in Features	59
4.4 Fractal Dimensionality Reduction	60
4.5 Statistical Entropy Relevance Estimation	63
4.6 Proposed Algorithm	65
4.6.1 Unsupervised Redundancy	66
4.8 Experimental Result	68
4.7 Discussion	94
4.9 Summary	95
<b>Chapter 5 Supervised Feature Selection</b>	<b>94</b>
5.1 Preliminaries	94
5.2 The Filter Approach	96
5.2.1 Filter Approach Based on MI	96

5.2.2 Maximal Relevance Minimum Redundancy (mRMR)	139
5.2.3 Supervised Redundancy	100
5.3 Algorithm Proposed	100
5.3.1 Algorithm Implementation	102
5.4 Experimental Result	103
5.5 Discussion	138
5.6 Summary	138
<b>Chapter 6 Conclusion</b>	<b>191</b>
6.1 Contribution	137
6.2 Conclusion	138
6.3 Suggestions for Future Research	139
<b>Appendix A Back-Propagation Algorithm</b>	<b>140</b>
<b>Appendix B Supervised feature Selection Algorithms</b>	<b>144</b>
<b>Appendix C PCA projections of the benchmark datasets used</b>	<b>146</b>
<b>Appendix D A two step Clustering Algorithm</b>	<b>140</b>
<b>References</b>	<b>165</b>

# LIST OF FIGURES

<b>Figure</b>	<b>Page</b>
2.1 Self-similarity in 1, 2 and 3 dimension. New pieces when each line segment is divided by M.	8
2.2 a) Sierpinsky pyramid, b) Sierpinsky triangle.	9
2.3 Illustration of box-counting method, in every step the size of the grid r is decreased in half to analyse the data in different level of resolutions.	10
2.4 Points generated to calculate the FD of an object.	11
2.5 Filter Approach	17
2.6 Wrapper Approach	18
2.7 Embedded approach	19
3.1 Diagram for relevant Features	33
3.2 DIANA	49
3.3 Artificial dataset, 80.15% of accuracy using 10 number of steps in the phase one of the fractal DIANA.	53
3.4 Artificial dataset, 90.14% of accuracy using 11 number of steps in the phase one of the proposed fractal algorithm.	53
3.5 Artificial dataset 97.94% of accuracy using 14 number of steps in the phase one of the proposed fractal algorithm.	54
3.6 a) Sierpinsky pyramid and, b) Sierpinsky triangle	61
3.7 Illustration of the calculation of the FI	65
3.8 Breast cancer Wisconsin unsupervised and supervised FS comparison	70
3.9 Breast cancer Wisconsin diagnostic (WDBC) unsupervised and supervised FS comparison.	71
3.11 Breast Cancer Wisconsin prognostic (WPBC) unsupervised and supervised FS comparison.	72
3.12 Australian credit approval unsupervised and supervised FS comparison	73
3.13 Echocardiogram unsupervised and supervised FS comparison	74
3.14 Ecoli unsupervised and supervised FS comparison	75

3.15	Statlog Heart unsupervised and supervised FS comparison	76
3.16	Hepatitis unsupervised and supervised FS comparison	77
3.17	Ionosphere unsupervised and supervised FS comparison	78
4.1	Iris unsupervised and supervised FS comparison	79
4.2	Lymphography unsupervised and supervised FS comparison	80
4.3	Parkinson (K-means) unsupervised and supervised FS comparison	81
4.4	Pima indian diabetes unsupervised and supervised FS comparison	82
4.5	Lenses unsupervised and supervised FS comparison	83
4.6	Glass identification unsupervised and supervised FS comparison	84
4.7	Tae unsupervised and supervised FS comparison	85
4.8	Mammography classes unsupervised and supervised FS comparison	86
4.9	SPECTF Heart unsupervised and supervised FS comparison	87
4.10	Vehicle unsupervised and supervised FS comparison	88
4.11	Vowe context unsupervised and supervised FS comparison	89
5.1	Wine unsupervised and supervised FS comparison	90
5.2	Zoo unsupervised and supervised FS comparison	91
5.3	Wood (real application) unsupervised and supervised FS comparison	92
5.4	Traditional framework of feature selection	101
5.5	Proposed framework of feature selection	101
5.6	Australian Credit approval SFS comparison	107
5.7	Echocardiogram SFS comparison	108
5.8	Ecoli SFS comparison	109
5.9	Heart SFS comparison	110
5.10	Hepatitis SFS comparison	111
5.11	Image Satellite (Statlog version) SFS comparison	112
5.12	Image Segmentation SFS comparison	113
5.13	Letter recognition SFS comparison	114
5.14	Ionosphere SFS comparison	115
5.15	Lenses SFS comparison	116
5.16	Mammography Masses SFS comparison	117
5.17	Parkinson SFS comparison	118
6.1	Shuttle SFS comparison	119
6.2	Sonar SFS comparison	120
6.3	Spambase SFS comparison	121

6.4	Vehicle SFS comparison	122
6.5	Breast Cancer (wdbc) SFS comparison	123
6.6	Wine SFS comparison	124
6.7	Pima Indian diabetes SFS comparison	125
6.8	Glass SFS comparison	126
6.9	Lymphography SFS comparison	127
6.10	SPECTF heart SFS comparison	128
6.11	Tae SFS comparison	129
6.12	Zoo SFS comparison	130
6.13	Breast Cancer Wisconsin Prognostic (WPBC) SFS comparison	131
6.14	Iris SFS comparison	132
6.15	Breast Cancer SFS comparison	133
6.13	Vowe Context SFS comparison	134
6.14	Wood (real application dataset) SFS comparison	135

# LIST OF TABLES

<b>Table</b>		<b>Page</b>
3.1	Benchmark datasets and artificial dataset used to test and compare the Fractal clustering algorithm, proposed and original version.	54
3.2	Experimental results using benchmark and artificial datasets.	95
3.3	Experimental results using the benchmark and artificial datasets used Table 3.2 for the FC algorithm proposed by Barbara (2003).	55
4.1	Description of benchmark datasets.	69
5.1	Description of benchmark datasets.	106

# ABBREVIATIONS

FD	Fractal Dimension
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
DNA	deoxyribonucleic acid
MI	Mutual Information
FS	Feature Selection
Jstor	Journal of the Royal Society
AI	Artificial Intelligence
SOM	Self-Organizing Map
DP	Dirichlet process
BIC	Bayes Information Criterion
AIC	Akiake Information Criterion
MML	Minimum Message Length
GMM	Gaussian Mixture Model
UFS	Unsupervised Feature Selection
EM	Expectation Maximisation
RIS	Ranking Interesting Subspaces
ID3	Iterative Dichotomiser 3
SBS	Sequential backward selection
SFS	Sequential forward selection
GSFS	Generalised Sequential Forward Selection
GSBS	Sequential backward Selection
PTA	Plus 1-Take Away
GPTA	General Plus 1-Take Away
GPTA	General Plus 1-Take Away
SBFS	Sequential Backward Floating Selection
SFFS	Sequential Forward Floating Selection
AFS	Adaptive Floating Search
OS	Oscillating search
SA	Simulated Annealing
GA	Genetic Algorithms

<b>PCA</b>	<b>Principal Component Analysis</b>
<b>FC</b>	<b>Fractal Clustering</b>
<b>DIANA</b>	<b>Divisive Analysis</b>
<b>FI</b>	<b>Fractal Impact</b>
<b>UFS</b>	<b>Unsupervised Feature Selection</b>
<b>FDR</b>	<b>Fractal Dimension Reduction</b>
<b>PFD</b>	<b>Partial Fractal Dimension</b>
<b>FS</b>	<b>Feature Selection</b>
<b>mRMR</b>	<b>maximum Relevance and Minimum Redundancy</b>
<b>UCI</b>	<b>University of California in Irvine</b>
<b>ANN</b>	<b>Artificial Neural Network</b>
<b>MLP</b>	<b>Multilayer Perceptron</b>



# NOMENCLATURE

$N$	Number of features
$D$	Data Space
$XY$	Random <i>variables</i>
$A$	<i>Data set</i>
$B$	Number of boxes
$r$	Length of the box side
$\ell$	Fractal Dimension
$\eta$	Number of object in a reduced version of the full set
$\eta(\delta)$	Number of boxes occupied with
$K$	Number of clusters
$S$	Microstates in a non-uniform distribution
$m$	Thermodynamic state
$N_n$	Number of boxes needed to cover an object
$I$	Number of instances
$S(r)$	Sum of points in each cell at each resolution $r$
$x_i$	Data point
$M$	Number of dimension
$E$	Entropy
$S$	Similarity measure
$D_{ij}$	Distance between instances
$\alpha$	Constant of similarity measure
$k^{th}$	$k^{th}$ dimension
$E_F$	Entropy function proposed
$R_i$	Fractal ratio to measure redundancy in the $i$ th feature
$\Delta pD$	Difference between partial and total fractal dimension
$T$	Original variable set of features
$f_i$	$i$ th feature in the data
$P(C)$	Probability of different classes
$H(C)$	Entropy of class C

$H(C/F)$  Conditional Entropy given the

$c$  classes

$I(A;B)$  Mutual information between two variables

$p$  p-value in the statistical set

# Chapter 1

## Introduction

This chapter presents a general background of methods, motivation and objectives adopted in this research. At the end, the structure of the thesis is outlined.

### 1.1 Background

Euclidean geometry has endured over the centuries because it provides a good basis for modelling the world, but there are many circumstances where it fails. An example of this is when in 1950 the English mathematician, Lewis Fry Richardson, needed to estimate the coast line of Great Britain. Richardson soon realized that the length of the coastline was indeterminate because it depended on the resolution with which the measurements were made. In order to describe non-Euclidean structures, in 1977 the mathematician Benoit Mandelbrot developed the concept of a dimension that had no integer values and was capable of performing multi-resolution analysis. The term given by Mandelbrot to the dimension was Fractal. Since 1977 FD has attracted attention as a practical tool in many branches of science where Euclidean distance is used. Scientific fields such as hydrology, geophysics, biology and communication systems are just some examples (Mandelbrot and Ness, 1968).

When researchers realized how well fractals mimicked nature, they started to explore how to measure and apply the FD in their investigations. The FD has been used as a method of quantifying the complexity of a wide variety of materials, objects and phenomena. For example the growth of tree structures; nerve cell growth and degeneration, osteoporosity, metal fatigue, fractures, diffusion and electrical discharge phenomena. The FD can measure the texture of a surface and is able to replace or augment Fourier methods by providing an index of surface roughness quantification (Borodich, 1997).

Image compression is another field in which fractals have been introduced and proven useful. Fractal image compression methods were developed by Michael Barnsley in

1987 when he set up his company, *Iterated Systems*. Most images exhibit a high degree of redundancy in their information content: for example most adjacent pixels will be the same or of similar colour or intensity. This redundancy can be used to repack the image so that it takes up less storage space (Barnsley, 1988).

Contemporary Computer Aided Design (CAD/computer aided manufacturing (CAM) theories and systems are well developed only for Euclidean analytical objects, lines, curves and volumes. However, there are still many types of objects such as flexible objects with a deformable geometry, and non-Euclidean geometrical objects, that cannot be modelled by the current CAD representation schemes. In this case, the FD can be used to mathematically define these types of products (Chiu et al., 2006).

## 1.2 Fractal Mining

The goal of data mining is to find patterns. Classical modelling approaches look for Gaussian patterns that often appear in practice. However, distributions such as Poisson and Gaussian, together with other concepts like uniformity and independence, often fail to model real distributions. Work done in data mining and pattern recognition shows how fractals can often reduce the gap between modelling and reality. FD is used in the following ways:

- To find patterns in a cloud of points embedded in the hyper space.
- To find patterns in time sequences useful in characterization and prediction.
- Graph representation e.g. social or computer networks.

The first is when multidimensional clouds of points appear in traditional relational databases, where records with  $N$  attributes become points in  $N$ -D spaces (age, blood pressure, etc), for example medical databases, 3-D brain scans (Arya et al., 1993) and multimedia (Faloutsos et al., 1994) databases. In these settings the distribution of the  $N$ -d points is seldom uniform. It is important to characterize a deviation to this uniformity in a concise way (Agrawal and Srikant, 1994) and to find accurate and short descriptions in the databases. The characterization of the data helps to reject some useless information and to provide hints about hidden rules.

The second task is when data changes through time. Time sequences appear very often in many of data applications. For example, there is a huge amount of literature in linear (Box et al., 1994) and non-linear forecasting (Castagli and Eubank, 1992) as well as in sensor data (Papadimitriou et al., 2003).

The third task is so general that it seems to be apparent everywhere, precisely because graph representation has a large number of applications, for instance in the world wide web which is probably the most impressive real network. So finding patterns, laws and, regularities in large real networks has numerous applications. Examples include: Link analysis, for criminology and law enforcement (Chen et al., 2003); analysis of virus propagation patterns on social/e-mail networks (Wang et al., 2000). Another application is the analysis of empirical data collections of DNA segments, called Networks of Regulatory Genes to indicate possible gene connections and interactions they have with proteins to perform their functions (Barbasi, 2002).

An extended enterprise is a type of network comprising interconnected organizations sharing knowledge and manufacturing resources. The individual organizations team up with one another so that their competitive advantage can be improved. Fractals are introduced into the construction of extended enterprises to enable simpler performance and transparency in terms of modality, information, functionality and time (Hongzhao et al., 2004).

### **1.3 Statistical Entropy**

Entropy is considered a form of information and can be used to measure uncertainty of random variables (Fast, 1962). Unlike energy, information is not conserved. However, an analogy between information and energy is interesting because they share several properties. Energy and information can exist in any region of space, can flow from one place to another, can be stored for later use, and can be converted from one form to another.

In the context of physical systems uncertainty is known as entropy. In communication systems the uncertainty regarding which actual message is to be transmitted is also

known as the entropy of the source. The general idea is to express entropy in terms of probabilities, with the quantity dependent on the observer (features in the case of data). One person may have different knowledge of the system from another one, and therefore would calculate a different numerical value for entropy.

Statistical entropy is a technique that can be used to estimate probabilities more generally. The result is a probability distribution that is consistent with known constraints expressed in terms of averages. This principle has applications in many domains, but was originally motivated by statistical physics, which attempts to relate macroscopic, measurable properties of physical systems. Physical systems, from the point of view of information theory, provide a measure of ignorance (or uncertainty, or entropy) that can be mathematically calculated.

#### **1.4 MI (Mutual Information)**

MI is a measure that shows the amount of information between two random variables  $x$  and  $y$ . In MI conditional entropy is used to calculate the reduction of uncertainty about the variable  $x$ , when  $y$  is known (Cover and Thomas, 2006). For example MI is used in text classification to calculate the amount of information between certain words and the rest of the document (Michel et al., 2008). MI is used in speech recognition and to find combinations of audio feature stream (Ellis and Bilmes, 2000). Other areas of application of MI are probabilistic decision support, pattern recognition, communications engineering and financial forecast (Fayyad et al., 1996). More recently, MI has been widely used in data mining due to its robustness to noise and to data transformations.

#### **1.5 Motivation**

Finding groups in data is a practical way to handle large amounts of data. Powerful mathematical and statistical concepts such as fractals, MI and entropy are tools that work well independently, which have never been combined before, for data mining.

A combination of these tools to simplify supervised and unsupervised data mining methods is addressed in this research as set out below:

- Divisive hierarchical clustering algorithms are not generally available in the literature and have rarely been applied due to their computational intractability. In this research a combination of hierarchical clustering and fractals is proposed to make divisive clustering more accessible than existing methods such as that developed by Kaufman and Rousseeuw (2005).
- To identify important features, when labels are not available, in a dataset is a problem without a direct solution. An unsupervised feature selection approach based on fractals instead of Euclidean geometry is developed to improve on a solution proposed in (Dash et al., 1997).
- Common supervised feature selection algorithms (Liu et al., 2008, Peng et al., 2005) rely mainly on heuristics tools such as search techniques and learning machines to find correlations at the cost of burdening the feature selection process. FD in combination with other tools can alleviate computational burden in a pure mathematical basis.

## **1.6 Aim and Objectives**

The overall aim objective of the research is to combine fractals, statistical entropy and MI techniques for the development of new algorithms, to simplify the detection of groups in data under an unsupervised and a supervised framework.

The specific objectives are summarised as follows:

- To develop a new simpler top-down hierarchical clustering algorithm that uses the FD as a similarity measure. The methodology gives the algorithm the capability to find clusters without any initialization step which is needed in previous fractal clustering algorithms.

- To develop an unsupervised Feature Selection (FS) algorithm that combines the FD and statistical entropy to detect useful features. The use of the FD reduces the complexity of the statistical entropy calculation and the removal of irrelevant features.
- To develop a supervised FS algorithm that combines FD and MI to maximize the relevance and minimize the redundancy of features simultaneously. The algorithm should select useful features without using any intermediate or heuristic subset evaluation step as is used in current FS frameworks.

## 1.7 Outline of the thesis

**Chapter 2:** This chapter gives an overview of classical FS approaches and clustering. Fractals, entropy and MI concepts followed by their applications in data mining are introduced.

**Chapter 3:** This chapter proposes a new clustering approach to find natural clusters in data. The algorithm uses the FD as a similarity measure among instances. The divisive approach used in the algorithm simplifies the analysis procedure of previous Fractal Clustering (FC) algorithms.

**Chapter 4:** In this chapter, a new Unsupervised Feature Selection (UFS) algorithm that ranks the features in terms of importance is proposed. The algorithm takes advantage of mathematical properties of the FD to reduce the complexity of the entropy function, to calculate relevant features under an unsupervised framework.

**Chapter 5:** In this chapter a new FS method that uses MI and FD is presented. The proposed method maximises the relevance and minimises the redundancy of the attributes simultaneously. The new method proposes a simpler framework for the evaluation of useful features.

**Chapter 6:** In this chapter, the conclusions and the main contributions of this thesis are presented. Suggestions for future research in this field are also provided.



# Chapter 2

## Background

This chapter presents an overall description of the basic concepts, FD, statistical entropy and MI that are used in this research. The chapter also introduces clustering and unsupervised and supervised feature selection approaches and gives recent examples of clustering and feature selection techniques.

### 2.1 Fractal Dimension

Most of the objects that are present in nature are very complex and erratic in having a Euclidean geometric structure. Mandelbrot proposed the concept of fractal in trying to address a model able to describe such erratic and imperfect structures. Since Mandelbrot, (1977) proposed the technique of FD to quantify structures, it has attracted the attention of mathematicians (Flook, 1996), computer engineers and scientists in various disciplines. Mathematicians introduced FD to characterize self-similarity to overcome the limitations of traditional geometry (Mandelbrot, 1982) (Schroeder, 1991). Engineers and scientists in the area of pattern recognition and image processing, have used the FD for image compression (Barnsley, 1988), image medical processing (Zhuang and Meng, 2004), texture segmentation (Chaudhuri and Sarkar, 1998), face recognition (Zhao et al., 2008), de-noising (Malviya, 2008), and feature extraction (Traina et al., 2000). Physicists, chemists, biologists and geologists have used the FD in their respective areas as well. Fractal theory is based on various dimension theories and geometrical concepts. There are many definitions of fractals (Mandelbrot, 1982). In this thesis, a fractal is defined as a mathematical set with a high degree of geometrical complexity. This complexity is useful to model numeric sets such as data and images.

### 2.1.1 Self-Similarity

One of the characteristics of fractals is self-similarity as shown in Figure 2.1 by Sierpinsky pyramid and triangle. This property defines the geometrical or statistical likeness between the parts of an entity (dataset) and the whole entity (dataset).

Self-similarity implies a scaling relationship when little pieces of an object are exact smaller copies of the whole object, i.e., smaller pieces of a fractal will be seen at finer resolution. Self-similarity specifies how the small pieces are related to the large pieces, thus self-similarity determines the scaling relationship. For example, consider a line segment to measure its self-similarity a line is divided into  $M$  smaller line segments, this will produce  $\eta$  smaller objects. If the object is self-similar each of the  $\eta$  smaller objects is an exact but reduced size copy of the whole object. The self-similarity  $d$  can be calculated direct from the equation

$$\eta = M^d \tag{2.1}$$


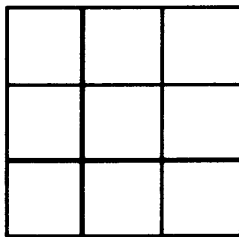
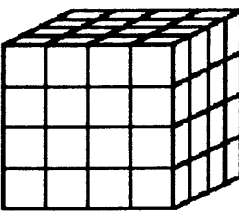
	$M$	$\eta$	$M^d$
 <span style="margin-left: 10px;">line</span>	5	5	$5^1$
 <span style="margin-left: 10px;">square</span>	3	9	$3^2$
 <span style="margin-left: 10px;">cube</span>	4	64	$4^3$

Fig. 2.1: Self-similarity in 1, 2 and 3 dimension.  $\eta$  new pieces when each line segment is divided by  $M$ .

Equation (2.1) is solve for  $d$  using logarithms properties on both sides of the equation as is shown next

$$d \log_2(M) = \log_2(\eta) \tag{2.2}$$

and finally equation (2.2) can be written as

$$d = \frac{\log_2(\eta)}{\log_2(M)} \tag{2.3}$$

To quantify the self-similarity of an object its FD needs to be calculated. FD describes how the object fills up the space, giving information about its length, area and volume. Its value can be an integer or a fraction.

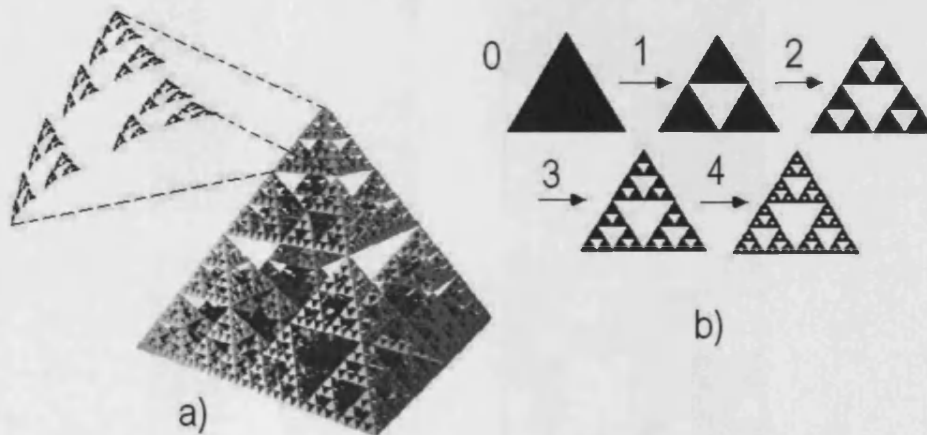


Fig. 2.2: a) Sierpinsky pyramid, b) Sierpinsky triangle.

Mathematically, the FD of a given set A is defined as follows:

$$\ell = \frac{\partial \log_2 B_n(A)}{\partial \log_2 r}, r \in [r1, r2] \tag{2.4}$$

where  $\ell$  is the FD of the set A.  $B_n$  denotes the number of boxes used to cover the object and  $r$  the length of the box side. The FD describes how an object fills up a space and gives information about the length, area, or volume of an object. The box-

counting method illustrated in Fig (2.3) is probably the most popular method due to its simplicity and good accuracy.

### 2.1.2 Box Counting Dimension

The box counting dimension is a useful way to measure the FD. The method consists in covering an object (data) with a grid and counting how many boxes the grid contains, for at least some part of the object. The counting process is repeated a certain number of times (resolutions), each time using boxes half the size of the boxes used the previous time. The box counting method calculates the FD dimension using the slope, shown in Fig (set number), of the plot  $\log(\eta(\delta))$  versus  $\log(1/\delta)$  where  $\eta(\delta)$  is the number of boxes occupied and  $r$  is the size of the box.

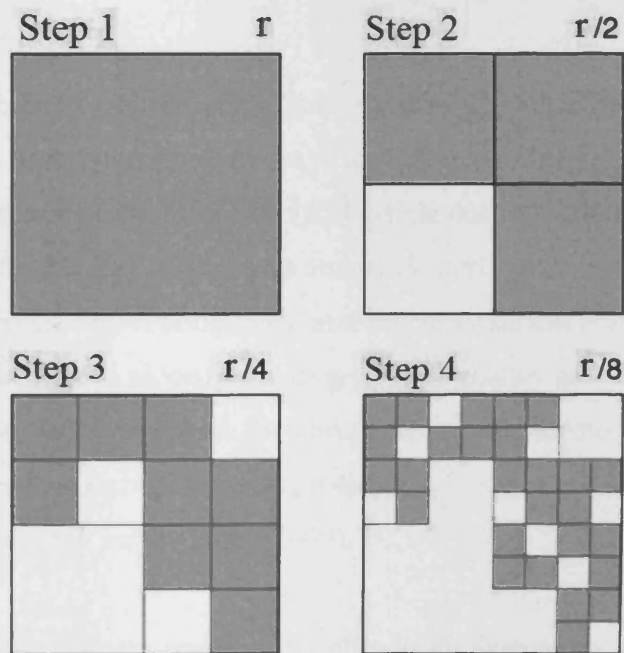


Fig. 2.3: Illustration of box-counting method, in every step the size of the grid  $r$  is decreased in half to analyse the data in different level of resolutions.

Each of the resolution in each step of the box-counting method generates a point like the ones show in Fig 2.4. The quantity for the first resolution corresponds to the first point up-right of the graph, and the last resolution corresponds to the last point bottom-left.

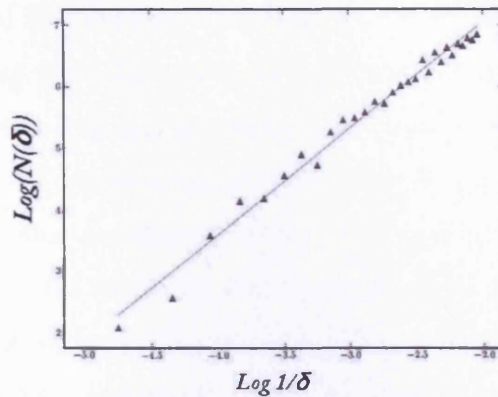


Fig. 2.4: Points generated to calculate the FD of an object.

If an object is perfectly self-similar the points generated in Fig 2.4 would create a perfect linear plot.

## 2.2 Clustering

Cluster analysis has its origins some thirty years ago when biologists and social scientists began to look for systematic ways to find groups in their data. The term data clustering first appeared in the title of a 1954 article dealing with anthropological data in a Journal from the Royal Society. The aim of clustering is to find a structure in data to generate hypotheses, detect anomalies, and indentify salient features. It can be used for natural classification to identify the degree of similarity among instances in data. Clustering has also been employed for compression as a method for organizing the data and summarising it through cluster prototypes. For all these reasons clustering is considered exploratory in nature (Jain, 2009).

The availability of computers made it possible to implement the resulting ways into algorithms, for different types of data. The most popular and simple clustering algorithm, k-means, was first published over 50 years ago. Thousands of Clustering algorithms have been published since then in a rich variety of scientific fields.

Nowadays clustering is applied in different fields, such as: Geosciences, political science, marketing, Artificial Intelligence (AI), chemometrics, ecology, economics, medical research, psychometrics. Since the eighteenth century researches such as Linnaeus and Sauvages have provided extensive classification of animals, plants,

minerals and diseases. In astronomy Hertzprung and Russell classified stars into various categories using their light intensity and their surface temperature. In social science for example behaviour and preferences are usually used to classified people. Marketing tries to identify market segments which are structured according to similar needs of consumers (Arabie and Hubert, 1994). In geography the objective is to group different types of regions. In medicine discriminating different types of cancer is one of the main applications as well as studying the genome data (Brandon et al., 2009). In chemistry classifying compounds and in history grouping archaeological discoveries, are other uses of clustering.

Image segmentation, a branch of AI, is an important problem in computer vision which is being formulated as a clustering problem (Jain and Dubes, 1988a, Frigui and Krishnapuram, 1999, Shi et al., 2000). Automatic text recognition can be efficiently performed using hierarchical clustering techniques (Iwayama and Tokunaga, 1995). Clustering is also used in planning group services to deliver engagements for workforce management (Hu et al., 2007).

### **2.2.1 Clustering Techniques**

Clustering methods can be broadly classified into two categories: partitioning and hierarchical methods (finding groups in data). In partitioning methods the basic idea is to construct  $k$  clusters. In each cluster there is at least one object and each object must belong to exactly one of the clusters. To satisfy this, there must be at least as many objects as there are clusters. In partitioning methods the number of clusters is given by the user. Not all values of  $k$  lead to a “natural clustering”, so it is suggested to run the algorithm a few times using different values for  $k$ . A different option is to leave the computer to decide the number  $k$  by trying many possible values and choosing the one that is best at satisfying a particular condition.

Partitioning methods try to find a “good” set of clusters in which the members in each are close or related to one another, and the members of different clusters are far apart or very different. The goal is to uncover a structure that is already present in the data. The most popular partitioning methods are  $k$ -means and  $k$ -medoid which determine  $k$

cluster representatives (centroids) randomly, and assign each object to the cluster with its representative closest to the object. The procedure iterates until the distances squared between the objects and their representatives is minimized. Extension of k-means and k-medoids for large databases is CLARANS and BIRCH (Zhang et al., 1996). Other types of partitioning methods are those which identify clusters by detecting areas of high density object population. DBSCAN (Ester et al., 1996) finds these regions, which are separated by low density points, by clustering together instances in the same dense region. Self-Organizing Map (SOM) is an effective clustering algorithm (Kohonen, 1982) that can converge into optimal partitions according to similarities in the data. The SOM performs with prototype vectors that themselves can be associated to each cluster centroid. There are different versions of the SOM algorithm that try to overcome the measure of similarity among objects (Alahakoon and Halgamuge, 2000).

Hierarchical methods are among the first techniques developed to perform clustering. Unlike partitioning methods, hierarchical methods do not divide the data in a determined number of clusters. In theory they deal with all the values of  $k$  from one to  $n$  which is the number of elements in the data. When the partition has  $k=1$  clusters, all the instances are together in the same cluster, and when the partition has  $k=n$  every instance is considered a cluster. The only difference between  $k=r$  and  $k=r+1$  is that one of the  $r$  clusters is divided to obtain  $r+1$  clusters. Note that hierarchical algorithms can be used to generate a partition by specifying a threshold on the similarity of the instances.

There are two types of hierarchical techniques. The first is called agglomerative, and the second, divisive. They construct their hierarchy in opposite directions, usually causing different results. The agglomerative approach builds clusters by merging two smaller clusters in a bottom-up mode. All the clusters together form a tree where leafs are the generated clusters and the root is the group with all the data instances. The divisive approach splits a cluster into two smaller ones in a top-down mode forming also a tree, the root in this case is the opposite to that in the agglomerative approach. However the tree structure is not exactly the same and depends crucially on the criteria used to choose the clusters to merge and split.

Hierarchical approaches have the disadvantage that they cannot correct what has been done in previous steps. In fact, once the agglomerative approach has joined two clusters, they cannot be separated any longer. In the same way when a divisive algorithm has divided an object, it cannot be reunited. However this limitation, in both hierarchical methods, is a key to achieve small computational times. On the other hand, hierarchical techniques do not really compete with partitioning approaches as they describe the data in a totally different way. The most well-known hierarchical algorithms are single-link and complete-link. Xiong et al., (2009) proposed a hierarchical divisive version of k-means called bisecting k-means, that recursively partitions the data into two clusters at each step.

### **2.2.2 Number of clusters**

To determine how many clusters are in a dataset has been one of the main problems in clustering. Very often, clustering algorithms are run with different values of number of clusters ( $K$ ); the best value for  $K$  is selected based on a predefined criterion. It is not easy to determine automatically the number of clusters, the task casts into the problem of model selection. The Dirichlet Process (DP) (Ferguson, 1973) (Rasmussen et al., 2009) proposes a probabilistic model to derive a posterior distribution for the number of clusters, from which the most likely number of clusters can be computed. Other techniques to calculate the number of clusters are Bayes Information Criterion (BIC) and Akaike Information Criterion (AIC). (Tibshirani et al., 2001) use gap statistics assuming that when the data is divided into an optimal number of clusters, the partition is more resilient to random perturbations. Figueiredo et al., (2006), Jain and Flynn, (1996) use the MML (Minimum Message Length) criteria together with GMM (Gaussian Mixture Model). This approach starts with a large number of clusters which are gradually reduced if the MML criterion is minimized.

### **2.2.3 Cluster Validity**

Cluster validity procedures intend to evaluate qualitatively and objectively the results of the cluster analysis. There are three basic criteria in which cluster validity can be performed: internal, relative, and external (Jain and Dubes, 1988). The internal



criterion compares the fit between the structure generated by the clustering algorithm and the data using the data alone. Hirota and Pedrycz (1985) propose a way of evaluating the fuzzy clustering methods using probabilistic sets and entropy characterisation. The relative criterion compares multiple structures that are generated by the same or by different algorithms, and decides which one of them is better. The external criterion measures the performance by matching the cluster structure to a priori information namely the class labels.

There is no best clustering algorithm. Each cluster algorithm imposes a structure on the data either implicitly or explicitly. When there is a good match between the model and the data, generally good partitions are obtained. Since the structure of the data is not known in advance, several approaches need to be tried before determining the best clustering algorithm for the application at hand. The idea of no clustering algorithm is the best is partially captured by “the impossibility theorem”. This theorem states that no single clustering algorithm simultaneously satisfies three basic axioms of clustering. The first is scale invariance, i.e. an arbitrary scaling of the similarity metric must not change the clustering results. The second axiom is richness, i.e. the clustering algorithm must be able to achieve all possible partitions of the data. The third is consistency, i.e. shrinking within cluster distances and stretching between-cluster distances, the clustering results must not change (Kleinberg, 2002).

The development of a top-down hierarchical clustering algorithm is presented in chapter four. The algorithm uses FD as a similarity measure. The validation of the clusters is done using the class label (Dash et al., 1997).

## **2.3 Feature Selection Approaches**

The objective of FS is to find a set of features of a certain size that provide the largest generalization and stability in data predictions. This has been mainly performed by selecting relevant and informative features that increase the efficiency of existing learning algorithms. FS is one of the central problems of machine learning so many algorithms incorporate the FS step in their framework. FS can have some other motivations such as:

- General data reduction - to limit storage requirements and increase algorithm speed.
- Feature set reduction - to save resources in the next round of data collection or during its utilization.
- Performance improvement - to gain in predictive accuracy in the predictions.
- Data understanding - to gain knowledge about the process that is generated by the data or simply to visualize the data.

A critical aspect of FS is to properly assess the quality of the features selected:

- Evaluation criterion definition (relevance index or predictive power).
- Evaluation criterion estimation (or assessment method).

Two of the main frameworks developed for feature selection are filters and wrappers. Both frameworks differ mainly in the way they evaluate the quality of the features. Filters uses evaluation criteria that does not employ feedback from learning machines for example criteria such as relevance index based on correlation coefficients or test statistics are examples of these criteria. Wrappers on the other hand use the feedback of the learning machine to assess the quality of the features. There is a third approach for feature selection called the embedded method which incorporates the feature subset selection and evaluation in the learning machine (predictor) (Guyon et al., 2006).

### **2.3.1 Filter approach**

The filter method, termed in (John et al., 1994), forms part of feature selection algorithms that are independent of any learning method. Such a method performs without any feedback from predictors and uses the data as the only source of

performance evaluation. These characteristics make the filter method the least computationally expensive approach and the least prone to overfit the learning machine.

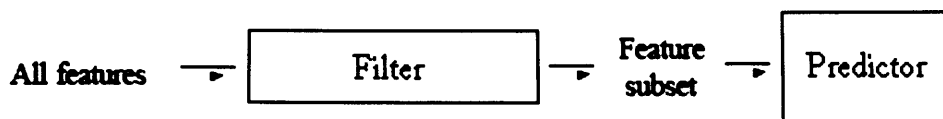


Fig. 2.5: Filter Approach.

The filter approach has a framework in which FS is performed as a pre-processing step for learning. In this framework there is no connection between the learning machine and the FS algorithm during the feature evaluation process. This characteristic gives to filters a remarkable superiority in terms of simplicity over other FS frameworks. Two of the most famous filter methods for feature selection are RELIEF and FOCUS (Kira and Rendell, 1992). In RELIEF the final subset of features is not directly selected, but rather each of the features is ranked in terms of its relevance to the class label. This FS method is ineffective at removing redundant features as the algorithm determines that a feature is important even if it is highly correlated to others.

The FOCUS algorithm guides an exhaustive search through all the feature subsets to determine a reduced set of relevant features. The search technique criterion makes FOCUS very sensitive to noise and to missing values in the training data. Moreover, the exponential growth of the number of possible feature subsets makes this algorithm impractical for domains with more than 25-35 features. The wrapper methodology proposed in (Kohavi and John, 1997) offers a simpler and more powerful way to search the feature subset space as it relies on a learning machine to evaluate the relevance of the features.

### 2.3.2 Wrapper approach

The wrapper approach is the second type of feature selection method. This approach receives direct feedback from a learning machine to evaluate the quality of the feature subsets. The machine learning is directly connected to the wrappers in order to drive

the search and train the learning model with all the subset of features. It is because of its structure that wrapper approach requires a higher computational effort to find an optimal subset of features.

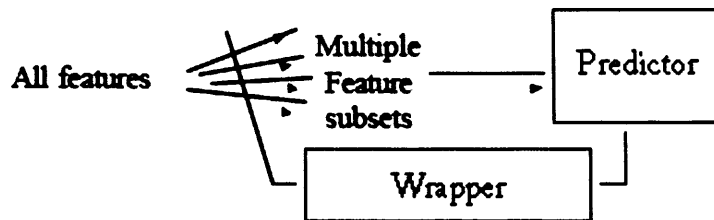


Fig. 2.6: Wrapper Approach.

Traditionally researches have been concerned about finding the best subset of features; in which the error estimation procedure yields a value no larger than for any other subset. But in the absence of a tractable strategy to do so, one which is able to guarantee the best subset, a useful strategy has to be considered.

In order to improve the tractability of wrappers, simpler search techniques are needed. Some of the simplest and most popular search techniques proposed by researchers are: greedy backward elimination, forward selection and nested (Guyon et al., 2006). These search techniques suffer from two main drawbacks: the first one is a tendency towards a sub-optimal convergence. The second drawback is the inability to find possible interactions among attributes. Despite these limitations, wrappers tend to produce better results in terms of classification accuracy.

In the section 2.2 a wide review of search techniques that can be used together with filters, wrappers and embedded, is presented.

### 2.3.3 Embedded approach

The embedded method unifies both theoretical methods, filter and wrappers. It creates a specific interaction between the learning machines and the feature selection process. This interaction combines the two procedures into one single entity. The inclusion of the feature selection process into the classifier learning procedure means the embedded methods are specific to the type of classifier that are considered (Guyon et

al., 2006). In contrast, wrapper approaches are independent from the kind of classifiers used, since any learning machine can be used to measure the quality of the features. The main advantage of the embedded method is a reduced computational effort than the wrapper method as it avoids repeating the whole training process to evaluate every possible solution.

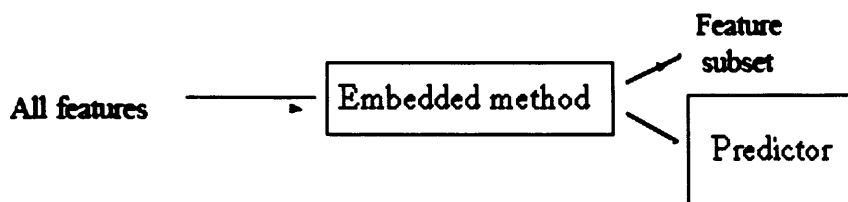


Fig. 2.7: Embedded Approach.

Each of the three feature selection approaches (filter, wrapper and embedded) has advantages and disadvantages. The filter approach is the fastest of the approaches as no learning is incorporated in the process of analysis. The wrapper approach, on the other hand, is the slowest one because in each iterations evaluates a cross-validation scheme. If the function that measures the quality of the feature subset is evaluated faster than the cross-validation procedure, the embedded method is expected to perform faster than the wrapper method. Embedded methods have higher capacity of generalization than filter methods, and are therefore more likely to overfit when the number of training samples is smaller than the number of dimensions. Thus, filters are expected to perform better when the number of training samples is limited.

All of the three previous approaches show difficulties in defining a relevance criterion (a relevance index for the performance of a learning machine) which has to be estimated from a limited amount of training data. Two strategies are possible: 'in-sample' or 'out-of-sample'. The first one (in-sample) is the "classical statistics" approach. It refers to using all the training data to compute an empirical estimate. The estimate is then assessed with a statistical test to measure its significance, or with a performance bound test to give a guaranteed estimate. The second one (out-of-sample) is the "machine learning" approach. It refers to splitting the training data into a training set used to estimate the parameters of a predictive model (learning machine) and a validation set used to estimate the model predictive performance. Averaging the

results of multiple splitting (or cross-validation) is commonly used to decrease the variance of the estimator (Guyon et al., 2006).

## **2.4 Unsupervised Feature Selection Approaches**

The objective of UFS is to select relevant features to find natural clusters in data. Feature selection can occur within two contexts: FS or UFS. As explain previously the difference between the two contexts is that FS is used for classification purposes, and UFS is applied for clustering tasks. It is broadly accepted that a large number of possibly not useful features can adversely affect the performance of learning algorithms, and clustering is not an exception. While there is a large amount of work on FS, there is little and mostly no recent research on UFS. This can be explained by the fact that it is easier to select features in a supervised context than in an unsupervised one. The reason for this is that in supervised learning it is known a priori the learning goal. The case is not the same for the unsupervised context, where the lack of label makes the relevant features hard to determine.

To find relevant features in data when the labels are not available (UFS) has to be seen exclusively as a problem within the data itself. In order to solve this problem, researchers have focused on looking for search techniques, evaluation functions and a stopping criterion.

The three approaches used in FS, filter, wrapper and embedded can be employed for UFS. The three approaches in USF follow the same line of efficiency as in the supervised context. The superiority of wrappers has been noticed in works such as (Guyon et al., 2006). However, their essential characteristics of using greedy search procedures remain suggesting that the filter approach is a less computational expensive alternative.

Wrappers and filters are the two approaches mainly considered to perform UFS tasks. Wrappers are relatively easy to implement in the supervised context since there is an external validation measure available (labels). On the contrary, in the unsupervised

context labels are not available and the wrapper has no guidance for the learning steps required in its FS process.

The main advantage of wrappers is that, in some cases, they are able to achieve the same performance as filters using fewer features. The probable reason for this is that most of filters techniques do not eliminate redundant features. Redundant features are those features that will not provide any improvement on the selected subset (Talavera, 2005). Filters might be less optimal than wrappers, but they are still reasonably good in performance and much less computationally expensive. The filter approach employs a sort of criterion to score each feature individually and to supply a ranking list. This approach can be extremely efficient as it has a less complex scoring procedure than other approaches.

Assuming the goal of clustering is to optimize some objective function to obtain “good quality” clusters, it is possible to use the same function to estimate the quality of different feature subsets. Although labels, when they are available, can be used as an external measure to validate the discovering of clusters.

## **2.5 Unsupervised Feature Relevance**

Features are said to be relevant if they are able to give a good description of the elements in a dataset. The objective of clustering is to group similar elements together (Kaufman and Rousseeuw, 2005) to have a good description of them. Irrelevant features have no influence on forming distinct clusters, but relevant ones do. The similarity of the dataset depends on the correlation of the features. Most clustering algorithms assume that the features in the dataset are equally important for the clustering task. But contrary to this, different features have different impact in creating clusters. A relevant feature helps to create a cluster while an irrelevant one may affect negatively (Dash and Liu, 2000).

In (Dash et al., 2002) a filter algorithm is presented with an entropy measure to determine the relative relevance of features. The algorithm ranks the features in terms of relevance taking into account the whole data rather than just individual clusters.

Talavera (2005) developed a wrapper algorithm based on Expectation Maximisation (EM) to estimate the maximum likelihood. This algorithm performs the clustering while at the same time selecting the best subset of features. In a general way this is an iterative procedure which alternates between two steps: the expectation step, and the maximisation step. An agglomerative algorithm developed by (Talavera, 2005) considers as relevant features, those ones which present high dependency within the rest of the features. The algorithm at each step merges similar values of elements among all the features, encouraging cohesive clusters and determining the most relevant features for these clusters.

RIS is a method proposed in (Kailing et al., 2003) that ranks subsets of features according to their clustering structure. The criterion used to measure the quality of the subsets is a density-based clustering notion of DBSCAN (Ester et al., 1996). RIS performs a bottom-up navigation through sets of possible subsets accumulating information and ranking them in terms of importance. Baumgartner and Plant, (2004) proposed an algorithm called SURFING (SUBspaces Relevant For clusterING). This algorithm measures the relevance of a subset of features using a criterion based on its hierarchical clustering structure.

## 2.6 Statistical Entropy

The concept of entropy was firstly explained by Boltzman in 1872. The statistical entropy defined in Eq. (2.3) was initially derived only for the number of possible arrangements in a gas or mixture of gases (Fast, 1962). However, its validity is so general that in all cases, as yet investigated, it leads to the same results analogous to thermodynamics. The tendency of entropy to a maximum value according to Eq. (2.3) means nothing else but a tendency to a state of large number of microstates, i.e. the tendency to a more probable state

$$S = k \ln m \quad (2.5)$$

The number of micro-states in the most probable distribution can be written mathematically as  $S = k \ln g_{\max}$ . Applying this equation to a gas only counts those



micro-states in which the gas, macroscopically considered, is distributed uniformly over the available space. On the other hand, when using Eq. (2.3) one only counts the microstates which correspond to non-uniform distributions.

The statistical definition of entropy is often introduced using the connection between probability and entropy. The principle is that the entropy is a function of the number of different ways in which  $m$ , a thermodynamic state, can be realized:

$$S = f(m) \quad (2.6)$$

If two systems are considered  $A$  and  $B$  as one system  $AB$ , its entropy is:

$$S_{AB} = S_A + S_B \quad (2.7)$$

According to classical thermodynamics a reversible state is reached in an isolated system, as soon as the entropy is at the maximum.

### 2.6.1 Entropy in Data Mining

ID3 is a very popular induction algorithm that uses the entropy measure to find the necessary information to classify instances. ID3 builds a decision tree system selecting the attribute with the minimum entropy in every node to split the tree (Quinlan, 1986). C4.5 is the successor of ID3 and is one of the most widely used decision tree algorithms. C4.5 has been augmented to C4.5 Rules to convert a decision tree into a rule set (Quinlan, 1995).

Zhu et al., (2010) proposed a model-based approach to estimate the entropy to overcome the sparseness in gene expression data. The entropy on a multivariate normal distribution model of the data is calculated instead on the data itself.

Lee et al., (2001) presented an algorithm that divides the feature space into good non-overlapped decision regions for classification applications. The algorithm divides the feature space using the feature distribution information calculated by Fuzzy entropy.

Yao et al., (1998) proposed an entropy-based clustering method that calculates the entropy at each data point and selects the one with minimum entropy as the first clusters centre. The algorithm automatically calculates the number and location of the cluster centres. After this, all data points that have similarity are placed, within a threshold, in a chosen cluster centre.

Cheng and Wei (2009) proposed a new entropy clustering method using adaptive learning able to find natural boundaries in datasets and classify similar objects into subsets. The algorithm uses a similarity measure among data points based on the calculation of entropy between two instances (Dash et al., 1997).

Wang (2005) developed algorithms that address the problem of clustering under the framework of possibility theory based on entropy. The algorithm has a number of advantages over other entropy-based approaches such as the ones in (Karayannis, 1994, Li and Mukaidono, 1995, Lorette et al., 2000). The algorithm calculates automatically the number of clusters by repeatedly merging similar clusters. A resolution parameter is proposed that determines regions of high density in the dataset.

Jing et al., (2007) extended the k-means algorithm by calculating a weight of importance for each dimension in each cluster. This is achieved using weight entropy in an objective function, that is minimized during the k-means clustering algorithm.

Tsai and Lee (2004) proposed an entropy based approach in neural networks that improves the learning speed, size or recognition accuracy for classification of structures. Entropy has been successful in determining the architecture of conventional neural networks in (Bichsel and Seitz, 1989, Cios and Liu, 1992, Lee et al., 1999). To improve the performance of a classifier Wang and Sontakke (2005) introduced a new packet classification algorithm that hierarchically partitions rule-bases using entropy and hashing.

## **2.6.2 Entropy as Relevance Measure**

In principle, statistical entropy is a technique that can be used to estimate input probabilities by calculating a probability distribution expressed in terms of averages. Statistical entropy has applications in many domains. Started being used in statistical physics to relate macroscopic and measurable properties of physical systems at atomic or molecular level (Dash and Liu, 2000). In the case of physical systems, entropy is considered a measure of uncertainty. In communications systems the uncertainty regarding which actual message is to be transmitted is known as the entropy of the source. In general, entropy depends on the observer. A person may have a different knowledge of the system from another and thus may calculate a different value for the entropy.

Data has orderly configurations if it has distinct clusters, and disorderly and chaotic configurations otherwise. Entropy is low in orderly configurations, and increases in disorderly configurations (Fast, 1962). Dash et al., (1997) proposed an algorithm to measure the entropy content in a data for several feature projections in order to determine irrelevant features. Jang and Chuen-Tsai, (1995) presented an algorithm based on entropy for UFS. Firstly the data is separated appropriately in clusters. Secondly for each of the clusters the entropy for different sets of features is computed. At the output, a list of features useful for the description of the data is generated.

Entropy has also been proposed for clustering analysis. For example, Cheng and Wei (2009) proposed a clustering algorithm based on entropy to avoid parameters. The method uses entropy to measure the mean of the distances between data points and the clusters centres.

## **2.7 Search Techniques Overview**

Because of the difficulties with optimal strategies, researches have proposed more complex search techniques to find reasonably good features subsets (features in the context of FS) without exploring all of them. There are plenty of these suboptimal approaches and some of them are described next:

**Exhaustive search** is a simple method and the only which guarantees to find the best subset. It evaluates every possible combination of features. With  $n$  candidate variables there are  $2^n - 1$  subsets to go through, making impossible the evaluation of all of them in a practical amount of time.

**Branch and bound** is a strategy proposed in (Narendra and Fukunaga, 1997) that searches only part of the feature space. The strategy is based on the fact that once a subset  $S$  consisting of more than  $d$  variables has been evaluated, we know, thanks to the monotonicity property, that no subset of it can be better. Thus, unless  $S$  excels the currently best known subset  $S'$  of target size  $d$ , the subsets of  $S$  need not be evaluated at all, because there is no way the evaluation result for any of them could exceed the score of  $S'$ . However the algorithm still has an exponential worst case complexity, which may render the approach infeasible when a large number of candidate variables are available.

The Branch and Bound method is not very useful in the wrapper model, because the evaluation function is typically not monotonic, i.e. adding features cannot decrease the accuracy. While methods like the RBABM (Kudo and Sklansky, 2000) are able to relax the requirement slightly, the problem is that typical predictor architectures, evaluated for example using cross-validation, provide no guarantees at all regarding the monotonicity (Guyon et al., 2007).

### 2.7.1 Sequential Selection Algorithms

**SBS** (*Sequential backward selection*) is a sequential pruning of variables introduced by Marill and Green (1963). This method is the first search technique suggested for variable subset selection. SBS starts with the variable set that consists of all the candidate variables. During one step of the algorithm, remaining variables in the set are considered to be pruned. The results of the exclusion of each variable are compared to each other using certain evaluation function. The step is finished by actually pruning the variable whose removal yields the best results. Steps are taken and variables are pruned until a pre-specified number of variables are left, or until the results get not good.

**SFS** (*Sequential forward selection*) is a similar algorithm to SBS proposed in (Whitney, 1971). The algorithm starts with an empty set and continues adding features. In one step each candidate feature that is not yet part of the current set is added, after the set is evaluated. At the end of the step, the feature whose inclusion resulted in the best evaluation is leaved in the current set. The algorithm proceeds until a pre-specified number of features are selected, or until there is no more improvement in the results.

SFS executes faster than SBS as in the beginning of the search when both methods have a big amount of possible combinations to evaluate, SFS evaluates much smaller feature sets than SBS. It is true that at the end of the SFS much bigger sets are evaluated than in SBS even that in this stage of the search there are very few combinations left to be considered. SFS evaluates the features in the context of only those that are already included in the subset. Thus, it may not be able to detect a feature that is beneficial alone but combined with other features.

## 2.7.2 Second generation of Sequential Selection Algorithms

After the sequential selection algorithms became known, a plethora of new versions of these basic search strategies were proposed. Some examples of these are described next.

**Generalized Sequential Feature Selection** methods are simple generalizations of SFS and SBS which include or exclude a subset of  $g$  number of features at a time and evaluate a created set. These types of algorithms are called GSFS (Generalised Sequential Forward Selection) and GSBS (Generalized Sequential backward Selection). When there are  $n - k$  candidate variables to be included (excluded), this

results into  $\binom{n-k}{g}$  evaluations in one step of the algorithm. This is much more than

the plain  $n - k$  evaluations in SFS, even if  $g$  is only two. On the other hand, the algorithms do not take as many steps as SFS and SBS, because they select more variables at a time, thus getting to a specified number of features using a smaller

number of steps. However, the combinatorial evaluation of one step decreases the complexity in the overall number of steps.

**Backtracking during search** is a method that solves the nesting effect in SFS, SBS, GSFS or GSBS. Using any of these algorithms, if a feature is added or removed in one step it will never be removed or added later in the process. This creates a problem called the nesting effect: bad decisions made at the beginning of the search cannot be corrected later as the selected features are fixed in the subset without the possibility of removal.

Backtracking has been used to deal with the nesting effect. During the search an algorithm named PTA (Plus 1-Take Away) divides each step into two substeps. In the first substep, SFS is run to include  $l$  new features. The second substep consists of running SBS to exclude  $r$  features from those that have been already selected. A straightforward generalisation of the PTA is simply to run GSF and GSBS instead of SFS and SBS, and this is how Kudo and Sklansky (1999) describe the GPTA (General Plus 1-Take Away) algorithm. However Kittler (1978) took the generalization a bit further by running GSFS in such a way that every step is split in a predefined  $r$  number of steps, reducing computational complexity. If  $r$  is equal to one, the algorithm is reduced to the nongeneralized PTA algorithm.

**Beam Search** is a method in which more than one subset is considered to be selected. In certain occasions it may be useful to analyze several promising subsets without restricting the selection to only one of them. It may also be desirable to be able to later return to different feature subsets other than the one that was chosen as the best one. In (Aha and Bankert, 1996) a list of several good feature subsets is maintained, when the list has only one set the algorithm is reduced to basic sequential selection. On the contrary, if there is a list with more than one good subset the algorithm analyses the first and then proceeds to analyse the subsequent ones in the list.

**Floating search** was introduced in (Pudil et al., 1994) and there are two versions of this methodology. The first is the SFFS (sequential forward floating selection) and the second is the SBFS (sequential backward floating selection). The basic idea in both is to perform a backtracking in the same way as in PTA, with the difference that in each

step the number of substeps is not limited. These substeps continue for as many as you need in order to improve the results obtained in previous steps.

SFFS (SBFS) has two different and alternating stages. The first stage runs one step of SBS (SFS). The second stage performs SFS (SBS) in which continues for as long as the feature subset is the best in size. When the subset decreases in quality, the first phase takes place again. The original floating point search methods had a minor bug which was pointed out and corrected in (Somol et al., 1999). When the algorithm returns to the first stage might be the case that a subset with less quality than the previous ones is found. The flaw is that the algorithm analyses this less promising subset even when a better one has already been found. In addition, Somol et al. (1999) developed a more sophisticated versions of SFFS and SBFS. The version is named Adaptive Floating Search (AFS) which switches from GSFS to GSBS, or vice versa, to select the best features.

In floating search, when the amount of backtracking is large more time is taken but there is a good chance to select better features. In addition to this, the algorithm can be stopped at any time because it keeps track of the best feature sets. Acceptable results are often obtained even if the algorithm has not yet finished (Guyon and Elisseeff, 2003).

*OS (Oscillating search)* was proposed in (Somol and pudil 2000). The algorithm is initialized using SFFS or SFS to set a good guess of the optimal number of features  $d$ . The OS consists of what the creators call down-swing and up-swing. During these processes the algorithm searches for subsets that are smaller and bigger than  $d$  respectively. The amplitude of the oscillation decreases if an improvement comes up during the swing, otherwise it increases. In this way computing power is decreased by restricting the search in a reduced number of feature spaces.

**Compound operator** is a technique used in sequential methods to reduce the computational cost in the pruning of irrelevant features (Kohavi and Somerfield 1995). Sequential methods waste computational power as irrelevance in the same features is calculated repeatedly in different subspaces.

The compound operator technique combines first and second best feature candidate to produce the first compound operator. More operators are produced by allocating the next best candidates. Candidates are added continuously until the operators do not degrade the result.

*Stochastic search algorithms* perform with random components. These components involve changes in the variable set. The changes vary depending on a particular initialization of the algorithm. This property of variability helps the algorithm to be more robust and consistent when values in the data are missed along different applications.

Stochastic optimization has been used to develop several algorithms for FS. Two of the most popular ones are Simulated Annealing (SA) and Genetic Algorithms (GA). They were first suggested for variable selection in (Siedlecki and Sklansky, 1988).

SA simulates the process of cooling down a body by minimizing its energy until it becomes a crystal structure. This analogy between the minimization energy process and the search for a minimum state in a system was first proposed in (Kirkpatrick and Gelatt, 1983).

For FS applications SA is initialized randomly with a feature subset (high temperature). A small random change in the feature subset is introduced in all next steps of the algorithm. If the subset is better, the change is accepted. If the subset is worse, the change is accepted with a probability proportional to the change. At the beginning of the algorithm the probability of an adverse change is more likely to be accepted than at the end of the algorithm (low temperature). As the algorithm iterates, the temperature decreases very often when no improvements are found. This is the reason SA at the beginning, when the temperature is high, is able to escape easier from local optimum. Still, it is able to find the exact local optimum at the end of the search due to low temperatures. The temperature does not allow deteriorating changes to be made anymore.

**GA** is part of the stochastic optimization algorithms family, a comprehensive introduction can be found in (Michalewicz, 1992). While the idea of SA is based in



physics GA comes from biological evolution motivation, where the best individual has the better chances of survival. In GA the solution subset of features is called chromosomes, and a set of solution subsets is a population. Another two concepts in GA, extracted from the biological vocabulary, are mutation and crossover. When mutation is applied a random bit or several bits of the chromosome are flipped to create a new chromosome. In crossover an offspring of two chromosomes is obtained by cutting both at some position and swapping the tails. A new population is formed when some of the chromosomes are retained in the old population, producing new chromosomes by applying genetic operations on the old chromosomes. The better the chromosome is the higher the probability of being selected to the new populations, or as parents in genetic operations.

***Rapid randomized pruning:*** This method was proposed by (Stracuzzi and Utgoff, 2004) to be specially used when the amount of relevant variables is small. The basic idea is to compute the probability that an important feature has in a randomly selected subset that is considered to be pruned. A subset of certain size is chosen to enable the straightforward removal of few features with a high probability of being irrelevant. If the estimation of the error increases as a consequence of the exclusion, it is concluded that one or more of the pruned features were relevant. In this case, the removal is cancel and a new random subset is chosen. This process continues until many consecutive iterations fail. When the process takes so long might be a sign that all the remaining features are relevant.

## **2.8 Relevance in Features**

It is of great necessity to focus on the most relevant information when an overwhelming quantity and low quality data increases due to the massive generation of information of new technologies. From that point of view, preparing for learning can be divided into two subtasks: the first one, selecting which features are relevant in describing a principle and the second one, eliminating redundant ones.

In the machine learning literature there are a variety of definitions of feature relevance. The reason of this variety is that it always depends on the question:

“relevant to what”, more specifically, different definitions may be more appropriate depending on one’s goal (application). Relevance has been studied for over fifty years by (Keynes, 1921) who proposed an intuitive meaning related to the relationship between objects. In the case of supervised learning the idea is to be relevant to the pre-determined class. Kohavi and John (1997) suggest two levels of relevance:

The first level is called strong relevance. This category includes those features which must be retained by the feature selection process. If any of these attributes is removed the descriptive power of the feature subset,  $S$ , is negatively affected.

**Level 1** (strong relevance). A feature  $X_i$  is strongly relevant if there exists some  $x_i$ ,  $y$ , and  $S_i$  for which  $p(X_i = x_i, S_i = s_i) > 0$  such that

$$p(Y = y | X_i = x_i, S_i = s_i) \neq p(Y = y | S_i = s_i) \quad (2.8)$$

*Weak relevance* is the second level in terms of importance. This level includes features which are not essential to build a good feature subset. However, a weakly relevant feature can still be useful especially in combination with other features (Yu and Liu, 2004).

**Level 2** (weak relevance). A feature  $X_i$  is weakly relevant if it is not strongly relevant and there exists a subset of features  $S_i'$  of  $S_i$ , for which there exists some  $x_i$ ,  $y$ , and  $S_i'$  with  $p(X_i = x_i, S_i' = s_i') > 0$ , such that

$$p(Y = y | X_i = x_i, S_i' = s_i') \neq p(Y = y | S_i' = s_i') \quad (2.9)$$

A feature is relevant if it is either weakly relevant or strongly relevant otherwise, it is irrelevant (Kohavi and John, 1997b).

The third level of importance is called irrelevance, and includes features which are not useful at all.

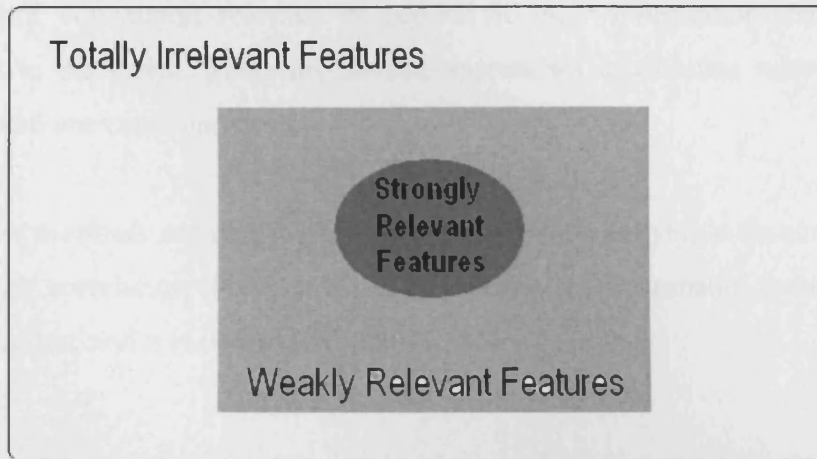


Fig. 2.8: Diagram for Relevant Features.

### 2.8.1 Relevance Approaches in FS

There are two types of FS approaches to detect relevance in features: univariate and multivariate. The univariate methods measure the individual relevance of the features and are considered faster and less prone to overfit. The multivariate methods which search for relevant subsets of features maintain the characteristics of wrappers and embedded FS methods.

A major disadvantage of the univariate methods is that they assume independence among features. The independence among features acts in two ways which are explained next:

- Features that individually are not relevant may become relevant in the context of other features.
- Features that individually are relevant may not all be useful because of possible redundancies.

Both multivariate and univariate FS methods typically focus only on measuring the relevance to the target. They keep relevant features and they eliminate the irrelevant ones.

Features are considered relevant in general if their information distribution is dependant to the target. There are several approaches to estimate relevance index, some of them are explained next:

**Correlation methods** are perhaps the simplest approach for single feature relevance. The Pearson correlation (Press et al., 1988) coefficient is probably the most classic relevance index, and it is defined as follows:

$$C(j) = \frac{\left| \sum_{i=1}^m (X_{i,j} - \bar{X}_j)(Y_i - \bar{Y}) \right|}{\sqrt{\sum_{i=1}^m (X_{i,j} - \bar{X}_j)^2 \sum_{i=1}^m (Y_i - \bar{Y})^2}} \quad (2.10)$$

where  $X_j$  is the  $n$  dimensional vector that contains all the values of the  $j^{th}$  feature and  $Y$  the  $n$  dimensional vector with all the target values. Pearson correlation is used to calculate redundancy among features and relevance to the class.

In a group of features the relevance grows within the correlation between class and features, and decreases within the growing of the correlation among the features (Ghiselli, 1964, Reynolds, 1977).

**MI** is one of the most popular techniques that can be used as a measure of information dependence between two variables. If two variables are independent, there is no correlation between them and the MI between them is zero. If the two are dependent, the MI between them is closer to one.

An advantage of using MI over correlation functions is that MI is able to measure general dependencies while the correlation measures can only measure linear dependencies. A second major advantage of MI is that it can be applied to both, categorical and numerical variables. Correlation techniques can only be applied to numerical variables.

**PCA** is a useful statistical technique to highlight similarities and differences among features to reduce the dimension by extracting new components without losing too

much information. The method uses simpler linear transformations that can ease the selection of the features.

**RELIEF** (Robnik-Sikonja and Kononenko, 2003) is an algorithm that weighs features by estimating how well they are able to describe data in comparison to those that are similar in value. Relief is based on the  $K$  nearest neighbours from the same class, and the same number of vectors of different classes. Relief has been designed to be robust with the presence of noise and missing values. Also, relief is considered a very successful algorithm due to its simplicity in multivariate probability distributions calculations.

**Probability Distribution** of the features can be used to measure their relevance to the target. There are several relevance FS algorithms based on this technique (Ogura et al., 2010, Pudil et al., 2002, Inza et al., 1999), one of the most popular was proposed by (Kolmogorov, 1998). The algorithm consists of calculating the difference between the joint and the product distributions of the features:

$$D_K(Y, X) = \sum_i \sum_{j=1}^K |P(y_i, x_i) - P(x_i)P(y_i)| \quad (2.11)$$

Where  $Y$  and  $X$  are two different distributions  $P(X)$  the probability distribution of one or both of the features. This function is the base for the MI calculation, a technique previously mentioned.

**Decision Trees** are top-down hierarchical partitioning techniques, useful in selecting relevant features. The benefit of using decision trees, for continuous values, is that they split the elements of the features in relatively pure bins, i.e. automatic discretisation is performed as the tree grows. This makes more accurate the calculation of probabilities such as  $P(x_i)$  and  $P(y_i|x_i)$  which are needed to calculate the MI and other indices (Duch et al., 2003). A very effective decision tree algorithm for feature ranking is 1R (Holte, 1993) as it creates only single level trees.

**C4.5** tree (Quinlan, 1986) uses information gained to determine the splits and to select the most important features which are the ones closest to the root node.

**CHAID** (CHi-squared Automatic Interaction Detector) decision tree algorithm (Kass, 1980) measures association between classes and feature values using  $\chi^2$  statistics, if you have  $X, Y$  variables the strength association is defined as follows:

$$\chi^2 = \frac{\sum_{ij} (M_{ij} - m_{ij})^2}{m_{ij}} \quad (2.12)$$

where

$$m_{ij} = \frac{M_i \cdot M_j}{m} \quad (2.13)$$

where  $m_{ij}$  represents the expected number of observations assuming  $X, Y$  independence.

**CART** (Classification And Regression Tree) algorithm (Breiman et al., 1984) sums the squares of the class probability distribution for a tree node using the *Gini* impurity index  $J_{Gini}(Y) = 1 - \sum_i P(y_i)^2$ . Each of the features is split into subsets with discrete values, the Gini indices are calculated for each feature. The gain is proportional to the average of the sum of squares of all conditional probabilities  $J_{Gini}(Y, X) = \sum_j P(X_j) \sum_i P(y_i | x_j)^2 \in [0,1]$  which gives a measure of the probability concentration useful for feature ranking.

## 2.8.2 Relevance versus Optimality

If a feature is relevant it does not imply that it has to be included in an optimal feature subset. In the same way when the feature is irrelevant does not imply that it should be excluded from the optimal feature subset. In other words, the two most relevant features do not create the best pair of features.

A feature that is relevant in combination with other features may become irrelevant individually. Another different case is when two individually irrelevant features become relevant when used in combination (Guyon et al., 2006).

The relevance and redundancy influence in determining optimal feature subsets makes FS a combinatorial problem. This is one of the reasons why multivariate methods looking for a good combination of features, achieve better results.

## **2.9 Feature Redundancy**

When two features are very similar to each other, the corresponding class-discriminative power would not change in great manner if a feature is removed. For this reason redundancy analysis is needed to create smaller feature subsets with the same or better discriminative power. A good and small feature subset contains features highly correlated with the class, yet uncorrelated with each other (Hall, 2000).

If two features are perfectly redundant and only one of them is used in the learning process, the performance is not affected. But when the redundancy is not perfect the features may complement each other, and information is gained when both features are used.

A concerning issue about multivariate methods is that they are prone to overfit. The problem is aggravated when the number of features selected is still too large compared to the number of instances. In this case it is recommended to use a filter method to filter out the least promising features before continuing with any learning process. Still, one wonders whether one could potentially lose some valuable features through the filtering process.

Traditionally multivariate methods are used to measure feature redundancy. The detection is done mainly using search techniques. Greedy methods (forward selection or backward selection) are the most popular techniques used in this task. Forward selection can yield better results than backward selection if the univariate approach is

used (Guyon and Elisseeff, 2003). The backward elimination approach usually performs better at the cost of selecting larger feature subsets. However for too short feature subset the performance of the learning may decrease abruptly (Guyon et al., 2006).

Incremental search methods (forward, backward) can be used in combination with different techniques to measure the redundancy among features.

**Pearson's Correlation Coefficient** (Reynolds, 1977) is a traditional statistical approach to find pairs of strong correlated attributes. The main disadvantages of using Pearson's is its weakness in the presence of outliers and the assumption of a Gaussian distribution of the data. These limitations may cause the analysis of Pearson's Correlation to fail in some application (Jing et al., 2007).

**Fast Correlation** is another technique for feature selection which is used to find the correlation. The evaluation criteria used within this method is called symmetrical uncertainty given by the following expression:

$$SU = 2 \frac{IG(X,Y)}{H(X) + H(Y)} \quad (2.14)$$

where  $H(X)$  is the entropy of a variable  $X$  after observing the variable  $Y$ .

FD is a mathematical concept that provides a way to quickly detect redundant features in a deterministic way. (Traina et al., 2000) developed an algorithm based on the FD that is able to detect redundant attributes. This algorithm was implemented and used to develop the algorithms proposed in this research. (Bhavani et al. (2008) proposed a filter based on the FD that is non-parametric and shows the relationship between dimensionality and a proper number of resolutions to calculate FD.



## 2.10 Mutual Information

The MI between two features measures the each other dependence, amount of information shared by them. MI is capable to successfully detect different types of dependencies. MI information is a non-parametric and non-linear technique that makes no assumption about the distribution of the data. When a search technique is combined with MI, groups of features with lower mutual inter-dependency can be found. Thus, MI is a very popular method to measure redundancy among features (Peng et al., 2005). A mathematical description of MI is given following:

We can start with the uncertainty of a random variable  $X$  that can be measured by the entropy, defined as

$$H(X) = -\sum_{x \in X} P(x) \log P(x) \quad (2.15)$$

Where  $p(x) = \Pr(X = x)$  is the probability density function of  $X$ . Similarly the joint entropy of two random variables  $X$  and  $Y$  is:

$$H(X, Y) = -\sum_{y \in Y} \sum_{x \in X} P(x, y) \log p(x, y) \quad (2.16)$$

Conditional entropy refers to the uncertainty reduction of a variable when another is known. If the variable  $Y$  is given, the *conditional entropy*  $H(X|Y)$  of  $X$  with respect to  $Y$  is:

$$H(X | Y) = -\sum_{y \in Y} \sum_{x \in X} p(x, y) \log p(x | y), \quad (2.17)$$

where  $P(x | Y)$  is the posterior probability of  $X$  given  $Y$ . From the previous definition, if  $X$  fully depends on  $Y$ , then  $H(X|Y)$  is zero. This means that no more information is required to describe  $X$  when  $Y$  is known. Otherwise,  $H(X|Y)=H(X)$  denotes that knowing  $Y$  will do nothing to observe  $X$ . To quantify how much information the two variables  $X$  and  $Y$  share the *MI*  $I(X, Y)$  is used and defined as:

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \quad (2.18)$$

From above  $I(X;Y)$  will be very high if  $X$  and  $Y$  are closely related; on the contrary  $I(X;Y)$  will be close to the value of zero when the values are unrelated (Battiti, 1994).

MI is a non-parametric and non-linear technique that makes no assumption about the distribution of the data. It is successfully capable of detecting different types of features dependencies without relying on transformations of the different variables (Peng et al., 2005). Due to these reasons MI is a popular method used in FS to characterise the dependency among the features and their class label.

(Li, 1990) makes an extensive comparison between MI and the *correlation function* to show the superiority of the MI to find general dependences in symbolic sequences. An algorithm based on greedy feature selection that takes both MI with respect to the output class and with respect to the already-selected features is proposed by (Battiti, 1994). A forward selection algorithm that stops itself automatically and uses the MI as a criterion to find optimal features is proposed by (Francois et al., 2007). (Michel et al., 2008) developed a multivariate approach, aimed to decode cognitive information from functional Magnetic Resonance images, able to detect non-linearities between the features and the label. (Sanchez et al., 2007) proposes an extended definition of the MI among fuzzified continuous variables to select optimal features and to obtain the most informative fuzzy partition of the data.

## 2.11 Summary

This chapter presents a general review of clustering, unsupervised and supervised FS techniques to provide relevant background information for the research reported in subsequent chapters.

## Chapter 3

# Divisive Fractal Clustering Approach

### 3.1 Preliminaries

This chapter presents a clustering algorithm that uses the FD as a tool to find natural clusters in data. The aim of this chapter is to present the FD as a useful tool for partitioning data into clusters that contain similar instances within themselves, and dissimilar instances within other groups.

Organising data into sensible groups is fundamental to understand and learn. This can be achieved using methods that study and perform automatically a cluster analysis. Clustering analysis groups objects according to their intrinsic characteristics or similarity when a class labels are not available. Fractals are used to perform clustering by detecting self-similarity in groups. Instances in data that belong to a certain cluster present a large amount of self-similarity and considerably less with respect to instances in other clusters. The main idea of FC is to group points within a cluster in such a way that none of the points in the cluster change the FD of the cluster by a large amount.

Barbara (2003) proposes a FC algorithm that requires an initialization step to create initial clusters and to start its execution. The initialization step is very important for good performance and requires very good quality initial clusters which have a direct impact in the quality of the clusters.

This chapter proposes a hierarchical FC algorithm which does not require an initialization step in order to start its execution. Instead, a divisive hierarchical algorithm that uses the FD as a similarity measure is proposed. The advantages and disadvantages of the proposed algorithm are discussed by comparing results with the algorithm proposed by Barbara (2003). The data used in the experiments is artificial and benchmark data.

The chapter is organized as follows: Section 3.2 describes the hierarchical clustering algorithms. In section 3.3, the classic fractal clustering approach is presented. In section 3.4 the proposed clustering algorithm is explained. Results are shown in section 3.5 and the work is summarized in Section 3.6.

## **3.2 Hierarchical Clustering**

Hierarchical methods are the simplest of the techniques for clustering and it was first proposed in 1951 in (Florek et al., 1951). A hierarchical method suffers from a defect that can never repair what was done in previous steps. Indeed once an agglomerative algorithm has joined two objects, they cannot be separated any more. Also, whatever a divisive algorithm has split up cannot be reunited. This rigidity of the hierarchical methods has both an advantage and disadvantage. The disadvantage is an inability to correct erroneous decisions. The advantage is that it leads to small computation times.

Hierarchical clustering algorithms have several other advantages as well: they are robust when it comes to inputting parameters, they are less influenced by cluster shapes, they are less sensitive to largely differing point densities of clusters, and they can represent nested clusters. Hierarchical algorithms do not really compete with partitioning methods because they do not pursue the same goal, as they describe data in a totally different way. The hierarchical methods do not actually create clusters, but compute a general hierarchical representation of the dataset. The hierarchical cluster structure is not unique. It depends crucially on the criterion of choosing the clusters to merge or to split.

There are two kinds of hierarchical techniques; the first technique is called agglomerative (bottom-up), and the second, divisive (top-down). These two modes of analysis offer advantages and disadvantages and can lead to differences in the hierarchical structure.

Agglomerative clustering (bottom-up) starts with details and then works its way up to large clusters. Quality in final clusters, often large clusters, could be affected by unfortunate decision in the first steps. On the other hand divisive clustering (top-

down) starts with the main chunks. In the first step it splits the data into two parts and then goes on by dividing them further into pairs of smaller parts. Because the large clusters are determined first, the final clusters are less likely to suffer from mistakes in the earlier steps. Moreover, one might even halt the divisive process at a stage where one is no longer interested in further splits.

On the other hand, divisive analysis poses some computational problems, at least in principle. Indeed, if the first step of the algorithm involves considering all possible divisions of the data into two subsets, it becomes unfeasible. Even for middle size datasets the divisive step is computationally prohibited due to the larger number of possible combinations (Kaufman and Rousseeuw, 2005). The clustering algorithm proposed in this chapter intends to alleviate computational burden by not performing any combinatorial analysis in the divisive analysis.

In relation to agglomerative hierarchical clustering plenty of algorithms have been proposed, some of the most relevant are mention next:

BIRCH (Zhang et al., 1996) is a hierarchical agglomerative algorithm which one of the main characteristics is its low computational complexity, it achieves an  $O(n)$  I/O (input-output linear) cost. BIRCH is a clustering algorithm that constructs a data structure called cluster feature (CF) tree by scanning the data only once. BIRCH performs poorly when the clusters are not spherical in shape. Whereas STING is a divisive hierarchical clustering algorithm, that divides the data space into grids. The hierarchical structure is built by exploiting the ancestor-descendant relation between the grids. The complexity of STING attains  $O(G)$ , here  $G$  denotes the number of grids in the lowest layer. STING is the most suitable for handling two dimensional data such as geographical data.

Guha et al. (2003) developed a robust agglomerative hierarchical clustering algorithm called *ROCK*. The algorithm uses a novel concept of links to measure the *similarity/proximity* between a pair of data points.

Dash and Liu (2000) carried out extensive empirical studies to show that, except for a number of top levels of the hierarchical tree, all lower levels agglomerate clusters

which are very small in size and close in proximity to other clusters. They named that characteristic the *90-10 rule*, it suggests that most levels from the bottom merge pairs of very small clusters separated by very small distances. Based in this rule Dash proposed a hierarchical algorithm that significantly reduces time and memory requirement of other hierarchical methods.

Rodrigues et al. (2007) proposed an algorithm for incremental clustering of streaming time series that constructs a hierarchical tree-shaped structure of clusters using a top-down strategy. The main idea is to split the cluster into two child-leaves, a diameter of the new clusters should be less or equal than the diameter of the parent node. If the diameter of the leaf is greater than its parent's diameter, then a previously taken decision would no longer reflect the structure of the data. The results obtained show that their performance is nearly as good as a batch divisive clustering on stationary time series.

Hulle and Gautama, (2004) proposed a hierarchical algorithm that uses topographic maps to estimate density areas at every level in a hierarchy. This criterion is used to determine a number of clusters and to divide data into new subsets to be analysed in a next level.

Pavan and Pelillo, (2003) suggested a new divisive hierarchical clustering approach based on an idea of varying a regularization parameter during the clustering process. The algorithm starts with a sufficiently large value which yields a unique large cluster that comprises all data. As the value decreases, the algorithm attempts to split large and incoherent (with dissimilar instances) clusters into smaller pieces.

Xiong et al., (2009) proposed a hierarchical clustering method that effectively divides and categorises data using Multiple Correspondence Analysis (MCA). MCA is a powerful factor analysis tool for categorical data which is widely used in the social and behavioural sciences. The splitting procedure of the algorithm consists of two phases; preliminary splitting and refinement. The preliminary splitting is based on MCA. A refinement, of each bisection, takes place like a reassignment step of the k-means algorithm with  $k=2$ , where  $k$  is the number of clusters.

A clustering algorithm that combines the strengths on both partitioning and agglomerative methods is proposed in (Laan and Pollard, 2003). In this algorithm the clusters are partitioned into two smaller clusters with an enforced ordering of the clusters. Steps to unify the two closest clusters into one cluster are used to correct errors made in previous partitioning steps.

Ding et al. (2001) proposed four new divisive selection criteria for hierarchical clustering; the average similarity, the cluster cohesion, avg-cohesion and temporary objective. Ding applies the criteria in datasets in which the number of clusters  $K$  is already known. The results obtained from the clustering algorithm are compared with hierarchical agglomerative clustering using four different linkage functions. The similarity measure  $W = (W_{ij})$  is applied on the linkage functions as they can be translated into similarity functions. The translated linkage functions are; complete linkage, average linkage, MinMax linkage.

Ding set several observations from comparisons between agglomerative and divisive methods. Such observations led him to the conclusion that the agglomerative method is much slower with a complexity of  $O(n^3 \log(n))$  comparatively with  $O(n^2)$  for the divisive method.

### 3.2.1 Agglomerative analysis

Hierarchical agglomerative clustering begins with one-point cluster and recursively merges the most similar pair of clusters. In several domains, hierarchical agglomerative clustering algorithms are able to yield best-quality results. However, this class of algorithms are characterized by a high complexity which reduces the size of the datasets that can be handled. In standard cases such complexity is  $O(dN^2)$ , where  $N$  is the number of objects in the dataset and  $d$  the cost of computing the distances between two objects. When  $d$  is either constant or very small in quantity, the complexity is simplified to  $O(N^2)$ .

In some contexts, however computing distances can be a very expensive task, such as in the case of complex comparison functions or high dimensional data i.e. distance

computation between long strings. The computation of all object-to-object distances dominates the overall cost of the clustering process, and so any attempt to improve performances should be aimed at saving distance computations effort (Nanni, 2005).

In the algorithm presented in this chapter a fractal agglomerative method proposed in (Barbara, 2003) is used as a, if needed as refinement step. Some examples in data could be very far from any cluster and can be either considered as noise or assigned to one of the clusters according to a certain criteria. If an instance remains without cluster the refinement step uses the FD to measure its similarity to all clusters and place it in the most suitable one.

There exist some traditional agglomerative hierarchical algorithms reported in the literature which have a complexity of  $O(N^3)$ , such as (Fisher, 1987) (Gennari et al., 1989) (Gowda and Diday, 1991) (Jain and Dubes, 1988). More recently, another agglomerative clustering algorithm with  $O(n \log n)$  complexity called Chameleon has been developed. This algorithm uses interconnectivity and relative closeness among instances to discover natural and homogeneous clusters.

### **3.2.2 DIANA (Divisive Analysis)**

Most hierarchical algorithms work using agglomerative analysis. The DIANA technique is a hierarchical clustering technique that works in the opposite way to an agglomerative method (Barbara, 2003). At each step, DIANA splits up a cluster into two smaller ones until all clusters contain only a single element. This means that a hierarchy is built  $n-1$  steps when the data set contains  $n$  objects.

In the literature, hierarchical divisive methods have been largely ignored. In fact when hierarchical clustering is mentioned it is often assumed that agglomerative clustering is meant. Most books on clustering pay little attention to divisive techniques, and software techniques do not include divisive algorithms at all. The main reason for this appears to be the computational aspect. In the first step of an agglomerative algorithm all possible fusions of two objects are considered, leading to a number of combinations given by



$$C_n^2 = \frac{n(n-1)}{2} \quad (3.1)$$

This number grows quadratically with  $n$ , which is large but the computations still feasible. On the contrary a divisive algorithm based on the same principle would start by considering all possible divisions of the data set into two nonempty subsets, which amounts to a number of possibilities given by

$$2^{n-1} - 1 \quad (3.2)$$

the number above grows exponentially and soon exceeds the current estimate of the number of atoms in the universe (Kaufman and Rousseeuw, 2005). Even for medium size datasets, a complete enumeration approach is not feasible.

Nevertheless, it is possible to construct divisive methods that do not consider all divisions. (Macnaughton et al., 1964) proposed an iterative procedure using an average dissimilarity between an object and a group of objects. Other divisive methods use a dissimilarity matrix as an input which are based on the optimization of a bipartition (Wang and Sontakke, 2005).

### 3.3 Fractal Clustering

Natural groups of data present a high degree of self-similarity which can be measured using the FD. By calculating the FD of the whole data and the FD of each of the instances it is possible to group self-similar elements of data which are likely to belong to the same clusters. Few clustering algorithms that use the FD to find natural clusters in data have been proposed and are briefly described below:

Barbara (2003) proposes a clustering algorithm that needs to be initialized by using a different clustering technique to find  $N$  initial clusters. After this initial step the FD of each of the clusters is calculated. The instances are assigned to the clusters which they least affect the FD of the clusters.

Tasoulis and Vrahatis (2005) proposed an algorithm that uses the FD together with the *k-windows* clustering algorithm (Prasad et al., 2003) to discover cluster centres more efficiently and to identify different regions within a single cluster.

Wang, (2005) proposed a fractal clustering method that uses the FD to cluster self-affine genes. The algorithm provides a very natural way of defining clusters that is not restricted to any particular cluster shape. The clusters are built in such a way that the data points in the same clusters are more self-affined among themselves than to the points in other clusters, although the clusters do not have to present perfect self-similarity.

### 3.4 Proposed algorithm

In this chapter a top-down hierarchical clustering is proposed that uses the FD dimension as a similarity measure. The algorithm has two steps. In the first step the algorithm partitions the data using DIANA approach and the FD as a similarity measure.

The pseudo algorithm to calculate the FD, using the box-count technique, proposed in (Traina et al., 2000) and used in this research is shown next:

**Algorithm:** Compute the *fractal dimension*  $D$  of a dataset  $A$   
input: normalized dataset  $A$  ( $N$  rows, with  $E$  dimensions/attributes each)  
output: *fractal dimension*  $D$

*Begin*

For each desirable grid-size  $r=1/2^j$ ,  $j= 1, 2, \dots, l$

For each point of the dataset

Decide which grid cell it falls in (say, the  $i$ -th cell)

Increment the count  $C_i$  ('occupancy')

Compute the sum of occupancies

$$S(r) = \sum C_i^2$$

Print the values of  $\log(r)$  and  $\log(S(r))$  generating a plot;

Return the slope of the linear part of the plot as the fractal dimension  $D$  of the dataset

$A$ .

*End*

The previous algorithm is used with DIANA to develop an algorithm that divides the data in such a way that maximizes the similarity within clusters and minimizes the similarity among relatively small clusters.

A theoretical description of the algorithm proposed in this chapter is given using the following toy example:

Consider an initial group of objects  $\{a, b, c, d, e, f, g, h, i, j\}$  that contain all the objects in the dataset. In the first step, the algorithm splits the data into two sub-groups  $\{a, b\}$ ,  $\{c, d, e, f, g, h, i, j\}$ . This is not done by considering all possible divisions, but rather by means of a comparative process of the FD of the whole data. Such a comparison is done by calculating a Fractal Impact (FI) of all instances in the data. The FI impact of the  $i$ th instance is obtained by quantifying how each of the instances affects the FD of the cluster. In order to calculate the FI the following formula is used:

$$FI(i) = | pfd(i) - FD | \quad (3.3)$$

where  $pfd(i)$  is the partial FD of the  $i$ th instance,  $FD$  is the fractal dimension of the whole dataset and  $FI(i)$  is the fractal impact of the  $i$ th instance. In this example the total number of FI calculations is 10,  $\{FI_a, FI_b, FI_c, FI_d, FI_e, FI_f, FI_g, FI_h, FI_i, FI_j\}$ . The partition of the cluster into two sub-clusters is defined by a threshold  $T$  defined by the mean of the total number of FI calculations. The instances with a value of FI above  $T$  create one of the clusters, in this case  $\{a, b\}$ . The instances with a value of FI below  $T$  create the second cluster  $\{c, d, e, f, g, h, i, j\}$ .

In the second step the algorithm finds which sub-groups are possible to split up again by calculating their FD. It is not possible for the FD to calculate very small sub-groups though because they do not have enough instances. The groups that are not large enough are left as sub-clusters to possibly merge with other sub-clusters in a second phase of the algorithm. Let assume that the sub-group  $\{a,b\}$  is found not to have FD due to the lack of instances, so it is not split any more and is left as a sub-cluster. The other sub-group  $\{c, d, e, f, g, h, i, j\}$  is considered to have enough

instances and is split up using the same criterion used in phase 1, generating other two sub-groups  $\{c, d, e, f\}$   $\{g, h, i, j\}$ .

In the third step the two sub-groups are found to have FD so they can both be split to generate sub-groups  $\{c, d\}$   $\{e, f\}$   $\{g, h\}$   $\{i, j\}$ . All these sub-groups are very small and it is possible to calculate their FD, so all of them are considered sub-clusters, stopping the algorithm in this stage.

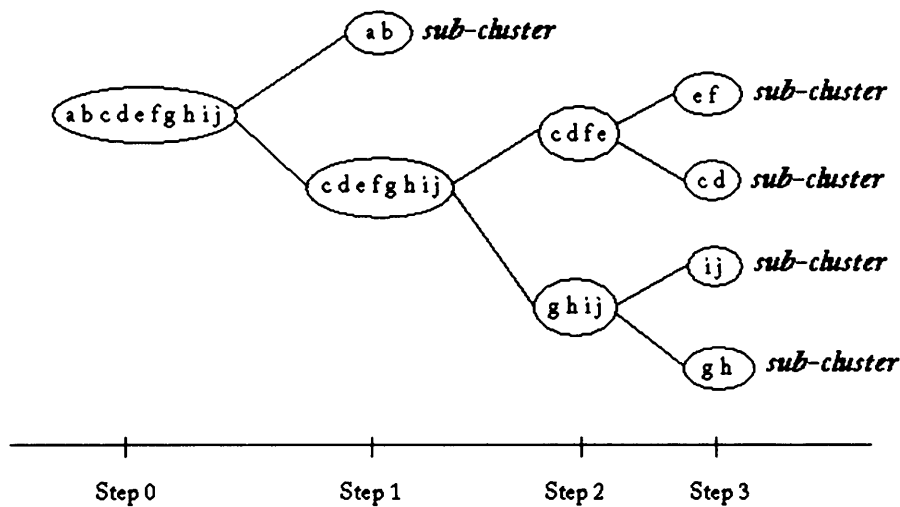


Fig. 3.1: DIANA

Crucial is how to best select the next sub-clusters to split or to merge. The fractal-DIANA algorithm, previously explained, would generate in a first stage relatively small sub-clusters. These sub-clusters need to be remerged in a second stage in order to build larger clusters.

### 3.4.1 Second Phase

The objective in the second phase of the algorithm is to find genuine clusters by repeatedly combining the sub-clusters that were found in the first step. The second phase of the algorithm merges sub-clusters that are close to each other to obtain the final clusters. The diameter of the cluster is the maximum distance between the variables of that cluster.

### 3.4.2 Stopping Criterion

The DIANA algorithm starts the clustering process dividing a cluster that contains all the instances in the data. The algorithm stops when single-points clusters, for each data point, are created. The fractal-DIANA algorithm starts the clustering process from the same single cluster that contains all the instances. The algorithm stops when all the sub-clusters created do not have FD and, in consequence, no more sub-clusters can be created.

### 3.4.3 Number of Clusters

The proposed algorithm does not address the problem of selecting the optimal number of clusters automatically. The number of clusters generated by the fractal-DIANA algorithm is set by the user. The clusters obtained, in the second phase, under the criteria of closeness and density, can vary depending on the number of sub-clusters obtained after the DIANA algorithm is applied. If the number of clusters is larger than the number of pre-defined clusters  $N$ , a re-merging phase is run. In this re-merging phase the  $N$  largest clusters are re-merged correspondently with close and small clusters that remain. All the clusters that are under certain threshold of proximity  $PT$  are merged to one of the  $i$ th large clusters. In this way all the small sub-clusters are merged to the  $N$  big clusters to create only the number of clusters set by the user.

The pseudo- algorithm of the clustering method proposed in this chapter is shown next:

---

**Algorithm** – *Divisive Fractal clustering algorithm (DFCA)*

input: dataset  $A$

output: all instances in the dataset organized in clusters

*Begin*

**Phase one**

- 1- Compute the fractal dimension  $D$  of the whole dataset;

- 2- Calculate the fractal impact of each instance in the dataset;
- 3- Divide the data generating two sub-groups according to the *FI* of each instance;

While *there are divisible sub-groups*

- 4- Compute the fractal dimension  $D$  of the  $i$ th sub-cluster;
- If  $(D_{ith} > 0)$  or  $(\text{sub-cluster size} > T2)$ 
  - 5- Calculate the fractal impact of each instance in the cluster;
  - 6- Divide the  $i$ th generated cluster;
- Else
  - 7- Store apart the  $i$ th sub-group and consider it as a sub-cluster

end

### ***Phase two***

If #sub-clusters  $>$  NC (number of clusters pre-defined by the user)

- 8 - Calculate the distances among sub-clusters
- 9 - Calculate the diameter of all sub-clusters
- 10- Select the  $N$  biggest sub-clusters and define them as clusters
- 11- Merge the remaining sub-clusters to their closest clusters

else

- 12 - Final clusters = sub-clusters

*End*

---

In the second step the proposed algorithm switches by repeatedly combining the sub-clusters, measuring their proximity with the Euclidean distance. The phase two of the algorithm involves merging the clusters that are the most similar and closest together,

### **3.4.4 Refining Step**

In the case that there are remaining instances finishing the first and second steps, the algorithm proposed in (Barbara, 2003) is run as a refining step to allocate the points in the proper clusters. After step two is finished, if some points are left without cluster assignment, a point that changes the FD of a cluster the least is assigned to it.

The clustering algorithm proposed adopts a common framework for clustering, which must discover cohesive and distinctive clusters. This framework assumes that a cluster must have similar feature values common to its members (cohesion) and few values common to member of other clusters (distinctiveness).

### 3.4.5 Evaluation Measure

The adaptation of the evaluation function is not straight forward, since most of the existing criteria relies on assessing how well a given feature subset discriminates among a set of predefined classes that are not available for unsupervised learners. There is no standard definition of irrelevance for an unsupervised prediction task. In the algorithm proposed the quality of the clusters is evaluated using the original labels of the data.

### 3.5 Experimental Results

In the following figures the results of the algorithm proposed on artificial data are shown. The data used is bi-dimensional and it contains 1000 instances. The data contain 3 clusters and is created using the *MATLAB fuzzy clustering toolbox* available in [www.mathworks.com/matlabcentral](http://www.mathworks.com/matlabcentral). The quality of clusters generated by the divisive fractal algorithm was evaluated measuring the accuracy within the original label in the datasets. The value of the means of the clusters found, by the proposed algorithm, and the means of the original clusters were used to define the cluster labels.

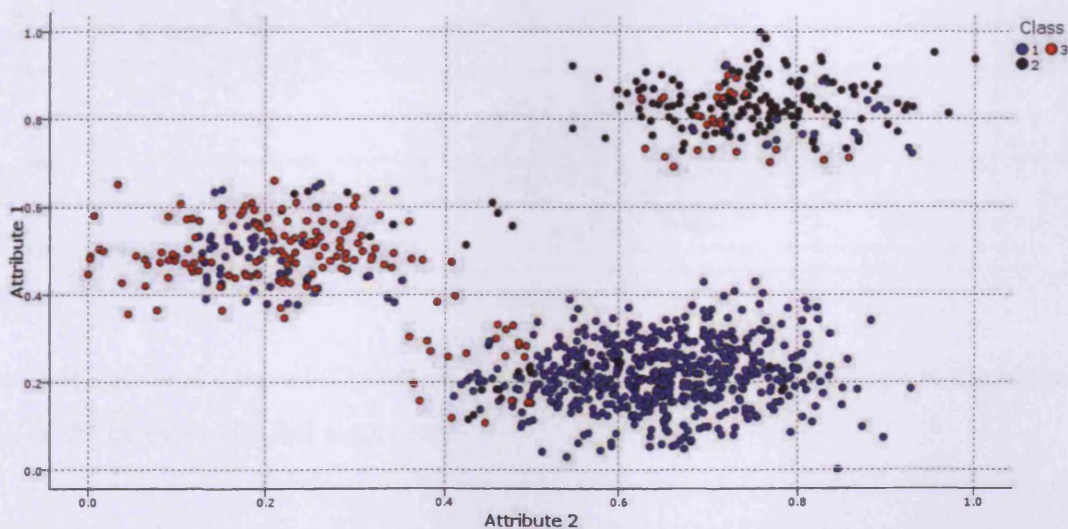


Fig. 3.2: Artificial dataset, 80.15% of accuracy using 10 number of steps in the phase one of the fractal DIANA.

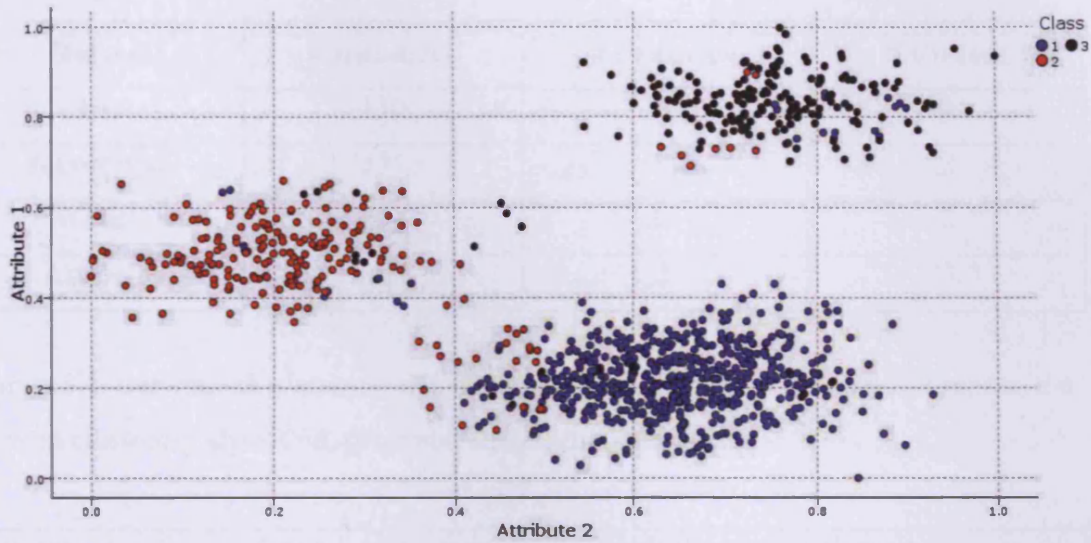


Fig. 3.3: Artificial dataset, 90.14% of accuracy using 11 number of steps in the phase one of the proposed fractal algorithm.

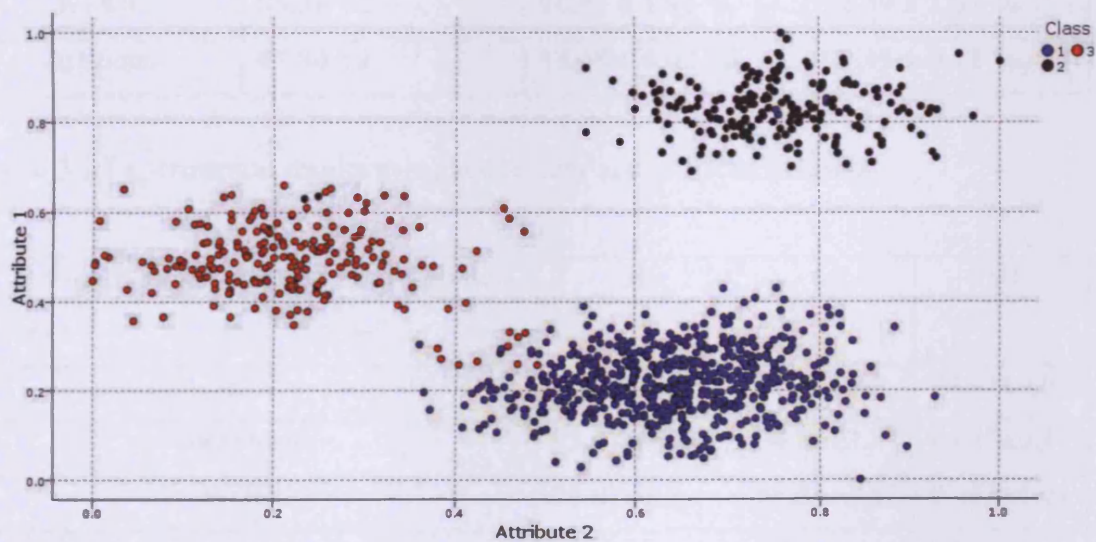


Fig. 3.4: Artificial data set 97.94% of accuracy using 12 number of steps in the phase one of the proposed fractal algorithm.

The artificial dataset above is partitioned until groups of data do not present FD. The DIANA partition reduces the data from 1000 to 252 groups of similar instances which are merged, with other subgroups, in the second stage of the algorithm. The fractal



DIANA performs the partitions for the best accuracy, as shown in Fig. 3.4, in 12 steps.

<b>Dataset</b>	<b># Instances</b>	<b># Features</b>	<b># Classes</b>
Iris	150	4	3
Ionosphere	351	34	2
WDBC	569	30	2
Artificial	1000	2	3

Table 3.1 Benchmark datasets and artificial dataset used to test and compare the Fractal clustering algorithm, proposed and original version.

<b>Dataset</b>	<b>Fractal Clustering (Proposed)</b>	<b>Hierarchical Clustering</b>	<b>K-means</b>
Iris	<b>83.6 %</b>	<b>84.15 ± 4.7 %</b>	<b>89.83 ± 3.51 %</b>
Ionosphere	<b>72.38 %</b>	<b>65.82 ± 11.3 %</b>	<b>68.88 ± 3.59 %</b>
WDBC	<b>85.31 %</b>	<b>90.86 ± 1.91 %</b>	<b>92.39 ± 1.67 %</b>
Artificial	<b>97.94 %</b>	<b>99.48 ± 0.02 %</b>	<b>99.95 ± 0.11 %</b>

Table 3.2 Experimental results using benchmark and artificial datasets.

<b>% of data used for initialization</b>	<b>30</b>	<b>60</b>	<b>90</b>
<b>Dataset</b>			
Iris	<b>82.35±0.0</b>	<b>87.22±0.0</b>	<b>94.41±0.0</b>
Ionosphere	<b>93.57±0.55</b>	<b>84.66±3.8</b>	<b>64.57±6.17</b>
WDBC	<b>93.81±1.5</b>	<b>85.54±5.17</b>	<b>87.53±1.41</b>
<b>%</b>	<b>75</b>	<b>80</b>	<b>90</b>
Artificial	<b>74.47±0.58</b>	<b>78.83±0.67</b>	<b>88.5±0.32</b>

Table 3.3 Experimental results using the benchmark and artificial datasets used Table 3.2 for the FC algorithm proposed by Barbara (2003).

### 3.8 Summary

In this chapter a divisive clustering algorithm that uses the FD as a criterion to find natural clusters in data is proposed. The algorithm performs a divisive methodology to find relatively small clusters (sub-clusters) in a first step. In a second step the small clusters are merged to find final larger clusters. This is done by comparing the whole FD dimension of each sub-cluster and the *pdf* of each instances within them.

The main difficulty in employing FD in clustering is that natural clusters exhibit self-similarity over a limited range of resolutions. The plots to calculate the FD are either a series of straight line segments or curved. A further limitation is that the FD provides a good index for the complexity of a boundary between two clusters but contains no information of the structure of that complexity, i.e. many different sub-clusters can have the same FD. The Fractal-DIANA algorithm proposed in this chapter is more suitable for approximately ball-shaped clusters.

# Chapter 4

## Unsupervised Feature Selection

### 4.1 Preliminaries

This chapter proposes an unsupervised feature selection algorithm based on a new fractal-entropy measure to rank features in order of importance. FD is a very efficient technique used to calculate similarity among objects. To simplify the detection of relevant features, in an unsupervised way, the use of the FD and entropy is considered in this research.

UFS is a way to simplify the discovery of clusters in data and most likely to improve their detection. The descriptive power of the data can be improved by selecting features that are relevant for a description process of the clusters. It is sometimes believed that larger datasets have better quality, but it is not rare for some of them to hide important structures and with this confuse the learning process.

An efficient way to handle large datasets and improve their description process is by selecting a subset of important features. Feature selection helps to understand the data better, to find clusters efficiently, to process, collect and store the data more efficiently. A meaningful feature helps to build clusters while a meaningless feature may have a negative effect in creating them.

Elimination of redundant features is important to reduce the size of the feature set. Even when you have a feature set full of relevant features some of them may be redundant as they do not add new information to the set. Various dependence measures like correlation coefficients, measures of statistical redundancy (Heydorn, 1971) and linear dependence (Das, 1971, Toussani and Vilmansen, 1972).

The number of features in a dataset makes the learning models more complex, like in the case of clustering. When UFS is performed this complexity is expected to decrease obtaining clusters with at least the same quality as the ones obtained using

all the feature set. Reducing the number of features used in the clustering process helps to deliver shorter cluster descriptions to the user. Shorter descriptions tend to be better understood hence they are less complex.

In this chapter, a new algorithm for UFS based on calculating entropy of data is proposed. The algorithm follows a new statistical entropy approach combined with the FD in order to detect relevant features for clustering. The statistical entropy measure proposed uses the FD instead of Euclidean distance to reduce computational complexity. The original Euclidean measure is reduced in complexity from quadratic to linear in terms of number of instances.

This chapter is organized as follows. Section 4.2 presents some classic and relevant work for unsupervised feature selection. In section 4.3, the concept of unsupervised relevance and redundancy are introduced. In section 4.4 the FD feature reduction is developed. In section 4.5 the classic approach of entropy to calculate relevance in features is presented. In section 4.6 the algorithm proposed is introduced. In section 4.7 results for the proposed algorithm on benchmark datasets and on a real application are given. Section 4.8 summarises the chapter.

## **4.2 Unsupervised Feature Selection**

Similar to a supervised approach there are two main unsupervised feature selection approaches. The first one is the unsupervised wrapper approach and the second one the unsupervised filter approach.

The wrapper method evaluates different feature subsets using an index measure and selecting the best one from the features subsets analysed. In the case of UFS, an index usually measures the quality of feature subsets without using a class label. A problem with the wrappers is their exponential computational cost in searching for useful features. On the other hand the advantage of using wrappers is their ability to commonly achieve an equal performance to filters with a more reduced dataset. Wrappers have an ability to make more accurate selection of relevant and non-redundant features. Unfortunately experiments provide evidence that suggests

wrappers are more prone to get trapped in local maxima, a common problem in sequential searching (Talavera, 2005).

In unsupervised approaches the wrapper methods evaluate the feature subset quality using a clustering algorithm such as k-means or EM algorithm as there is no unanimous criterion to estimate quality of features. Contrary to supervised FS, in which there is a consensual (label) way for assessing feature quality. The features selected by wrappers enable a specific clustering algorithm to find faster and better clusters. Unfortunately, the feature sets found by wrapper are not likely to work efficiently on different clustering algorithms. Another disadvantage of wrappers is that have the highest computational cost of all the three approaches.

Filter methods for UFS exclusively perform feature assessment on intrinsic properties of the data. Filters appear to be less optimal but they reasonably compromise the wrapper performance. Experimental evidence suggests that filters are able to perform a reasonably good job given the limited information they use. Features are selected without using any clustering method, this significantly reduces computational burden in comparison with other techniques such as wrappers.

Filters for UFS are still an uncommon approach in data mining. A notable exception is a work in (Dash et al., 2002, Dash and Liu, 2000, Dash et al., 1997) that proposes an unsupervised entropy measure for ranking features in terms of relevance. A problem that current filters methods present (Heydorn, 1971) is their sensitiveness to redundant features. To alleviate this problem some filter algorithms that use a search technique have been proposed.

Greedy algorithms such as sequential forward and backward search are very popular heuristic methods due to their simpler implementation. The sequential algorithms have quadratic complexity but perform poorly when the number of instances increases too much (Gheyas, 2009).

Dash et al, (1997) developed an unsupervised entropy-based measure in order to determine the relevance of the features. This method uses a new criterion based on statistical entropy. Entropy is low for data orderly configurations, and more for

disorderly configurations. The measure determines a feature to be relevant if removing the feature increases the entropy of the dataset. The algorithm searches for relevant feature subsets using a sequential backward selection method.

In literature most of the FS algorithms have been created for supervised learning. The absence of the class labels makes more difficult to find relevant features due to the lack of guidance they provide, and this might be the reason why there is not much research done in this area. Few and mostly recent UFS have been proposed for clustering. Most importantly, the majority of them are wrappers methods that require intense computational work. There are three categories for unsupervised feature selection algorithms according to their evaluation criterion in selecting useful features: filters, wrappers and embedded. Filters are the most efficient approaches to deal with high dimensional datasets.

### **4.3 Unsupervised Relevance and Redundancy in Features**

In a set of relevant features some might be redundant as they do not provide any additional useful information. Redundant features may deteriorate or not play an important role in generalising an ability to detect certain groups in the data. In addition, it is always desirable to find small and informative feature subsets, which also reduce computational effort, collection and storage requirements. UFS is not an easy task to perform as labels are not available to guide the selection process. The problem becomes even more challenging when the number of clusters is not known in advance.

The problem of redundant features in which the class label is not available has not been addressed in a great manner within the literature. The UFS literature usually accepts that the instances in a cluster have similar feature values, i.e. a cluster has members with redundant features. On the other hand, the presence of redundant features makes difficult to differentiate among clusters. Few works that consider both analyses redundancy and relevance have been proposed.

Zeng and Cheung, (2009) proposed a new feature selection method in which not only the most relevant features are indentified, but the redundant features are also eliminated so that smaller relevant feature subsets can be found.

Cord et al., (2005) proposed a wrapper UFS algorithm based on Laplace mixture model. The number of features selected uses a strategy based on the t-statistic to choose the most relevant features. After this, and evaluation of the clustering error to discard the redundant features from the relevant ones is applied. This strategy gives a good compromise between the selection of features and the performance of the clustering. This approach is computationally intensive as it is based on the wrapper method.

Li et al, (2007) proposed an algorithm that applies a sequential backward search able to find locally optimal subsets. i.e. different feature subsets for each of the clusters. This algorithm is based on the idea that different clusters may be better discovered in different subspaces. Regardless what the evaluation criteria is, global UFS can only find relevant feature subsets for all of the clusters together. The experimental result for this algorithm shows that the UFS outperforms global approaches on various datasets.

#### **4.4 Fractal Dimensionality Reduction**

Fractal theory is based on various dimension theories and geometrical concepts. There are many definitions of what a fractal is (Mandelbrot, 1977, Mandelbrot, 1982). In this thesis, a fractal is defined as a mathematical set with a high degree of geometrical complexity (Barnsley, 1988). The complexity quantified by fractals is useful to model and measure the complexity of numeric sets such as data and images (Tasoulis and Vrahatis, 2005). One of the characteristics of fractals is self-similarity. Fig. 4.1 shows two self-similar figures, the 2D and 3D Sierpinski set. Self-similarity defines geometrical or statistical likeness between parts of a set and the whole set. To quantify the self-similarity of an object its FD, defined by Eq. 4.1, needs to be calculated. This measure describes how the object fills up the space, giving information about its length, area and volume, the value of the FD can be an integer or fractional.

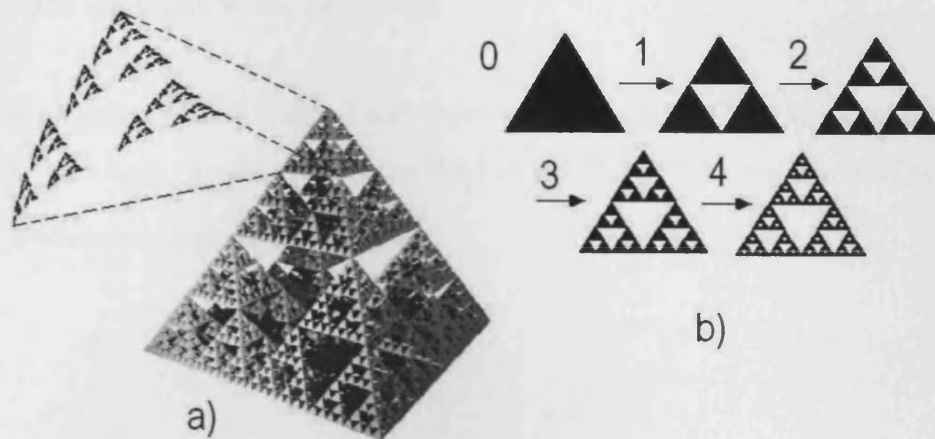


Fig. 4.1: a) Sierpinsky pyramid, b) Sierpinsky triangle.

Some fractal object has similar properties over certain range of sizes. Length, area and volume remain proportional at finer resolutions. Thus the value of the FD depends on the resolution used to make the measurement, giving you a different insight of how the object fills the space.

Mathematically the FD of a given set  $A$  is defined as follows:

$$FD = \frac{\partial \log N_n(A)}{\partial \log r}, r \in [r1, r2] \quad (4.1)$$

the FD of a set  $A$ .  $N_n$  denotes the number of boxes used to cover the object and  $r$  the length of the box side. Due to its relative simplicity and good accuracy, the box-counting method is the one chosen in this work to calculate the FD of a dataset.

FD has been used as a tool for the spatial access (Belussi and Faloutsos, 1995), indexing (Bohm and Kriegel, 1999), joint selectivity estimation and analysis of metric trees (Faloutsos et al., 1994).

An algorithm to calculate the FD based on the box-counting method, explained in chapter 2, is proposed in (Traina et al., 2000). The FD methodology is part of another algorithm able to quickly select redundant attributes in a dataset. The main idea of the



algorithm is to calculate the FD of the dataset and drop the features which do not affect it. The algorithm works as follows:

Consider a dataset with  $N$  features and  $I$  number of instances,  $(N \times I)$  size. Impose an E-grid with grid cells of side  $r$ . Focus on the  $i$ -th cell, let  $C_{r,i}$  be the count of points in each cell (occupancies). Then compute

$$S(r) = \sum_i C_{r,i}^2, \quad (4.2)$$

the derivative of  $\log(S(r))$  at different resolutions with respect to the logarithm of the radius is the FD of the whole data. The multi-resolution analysis is performed using a multi-level structure, where each level has a radius half of the size of the previous level ( $r=1/2, 1/4, 1/8\dots$ ). The different values of  $\log(S(r))$  versus the values of  $r$  create a two dimensional line graph, which slope is linear if the dataset is self-similar.

In order to detect and drop the redundant features (Traina et al., 2000) proposes a backward elimination reduction algorithm called FDR. The proposed idea is to calculate the FD of the whole dataset, and also the PFD dropping one of its  $N$  attributes at a time. There are  $N$  partial fractal dimensions, one for each attribute. The attributes whose PFD is very similar to the FD are considered the least redundant ones. The algorithm is able to detect linear and non-linear correlation among attributes and it is described next:

**Algorithm - Fractal dimensionality reduction (FDR) algorithm**

input: dataset  $A$

output: list of attributes in the reverse order of their redundancy

*Begin*

- 1- Compute the fractal dimension  $D$  of the whole dataset;
- 2-Initially set all attributes of the dataset as the significant ones, and the fractal dimension as the current  $D$  ;
- 3-While there are significant attributes do:
- 4-For every significant attribute  $i$ , compute the partial fractal dimensions  $pDi$  using all significant attributes excluding attribute  $i$ ;
- 5- Sort the partial fractal dimensions  $pDi$  obtained in step 4 and select the attribute  $a$  which leads to the minimum difference (current  $D - pDi$ );
- 6- Set the  $pDi$  obtained removing attribute  $a$  as the current  $D$ ;

7- Output attribute  $a$  and remove it from the set of important attributes;  
*end*

The computational complexity of previous algorithm grows linearly with the number of instances in the dataset and exponentially on the number of features.

#### 4.5 Statistical Entropy Relevance Estimation

A dataset is represented with  $j$  number of points in a  $M$  dimensional space. Where each data point  $x_i$ ,  $i = 1, \dots, N$ , is represented by a vector of  $M$  values. If the dataset has clusters included, it is said, it has an orderly configuration. If the data does not contain any cluster it is said it has a chaotic configuration. From (Fast, 1962) it is known that entropy is low for orderly configurations, and more for disorderly configurations. Thus, if the entropy for each of the projections can be measure it is possible to determine which of the features is useful to describe orderly configurations (clusters). Notice that in a dataset the entropy should be very low between two instances if they are very close or very far, and very high if they are separated by the mean of all distances. For two instances, the entropy measure is:

$$E = -S \log_2 S - (1 - S) \log_2 (1 - S) \quad (4.3)$$

where  $S$  is a similarity measure that is based on distances, and assumes a very small value (close to 0) for a very close pair of instances, and a large value (close to 1) for very distant pairs of instances. The variable  $E$  measures the entropy for two instances, that assume the maximum value of  $E=1.0$  when  $S=0.5$ , and the minimum value of  $E=0.0$  for  $S=0$  and  $S=1$ .

For a dataset of  $M$  dimensions the entropy of different feature subsets can be calculated and within it the relevance of each of the features. The statistical function used to measure entropy of  $I$  instances is as follows:

$$E = - \sum_{i=1}^I \sum_{j=1}^I (S_{ij} \log S_{ij} + (1 - S_{ij}) \log(1 - S_{ij})) \quad (4.4)$$

where  $I$  is the number of instances and  $S$  is a similarity measure defined as follows:

$$S_{ij} = e^{-\alpha D_{ij}} \quad (4.5)$$

where  $D_{ij}$  is the distance between the instances  $x_i$  and  $x_j$ .  $\alpha$  is a constant that mathematically is given as:  $\alpha = \frac{-\ln 0.5}{\bar{D}}$  where  $\bar{D}$  is the average distance among the instances in a hyperspace. Euclidean distance is used to calculate the distance  $D_{ij}$  between two data points  $x_i$  and  $x_j$ . In a multidimensional space it is defined as:

$$D_{ij} = \left[ \sum_{k=1}^M \left( \frac{x_{ik} - x_{jk}}{\max_k - \min_k} \right)^2 \right]^{1/2} \quad (4.6)$$

where  $\max_k$  and  $\min_k$  are the maximum and minimum values for the  $k^{\text{th}}$  dimension.

Dash et al. (1997) proposed a sequential backward selection algorithm to determine the relative relevance of features. In each iteration, the algorithm calculates the entropy  $E$  after removing one feature from the set of remaining features. A feature is removed as the least important one if a set without the variable gives the least entropy. This continues until the importance of all variables is determined. The complexity of the algorithm is  $O(M^2 N^2)$  where  $M$  is the number of features and  $N$  is the number of instances in the dataset.

The pseudo-algorithm to find important variables is shown next:

$M = \text{Total number of features}$

$T = \text{Original feature set}$

**Begin**

**For**  $k=1$  to  $M-1$  {iteratively remove features one at a time}

**For** every variable  $v$  in  $T$  {Determine which variable to remove}

$$T_v = T - v$$

Calculate  $ET_v$  on  $D$  using ec

Let  $V_k$  be the variable that minimizes  $ET_v$

$T = T - V_k$  {Remove  $V_k$  as the least important variable}

Output  $V_k$

**End**

## 4.6 Proposed Algorithm

In this chapter a UFS algorithm that uses a new entropy function to detect important features is proposed. The new entropy function  $E_F$  uses the FD to reduce one loop in the entropy function proposed in (Dash et al., 1997). The proposed function eliminates the calculation of distances between instances  $x_i$  and  $x_j$ . The Euclidean distance measure is not used to calculate the similarity among instances. Instead, the similarity is calculated using the FI of each instance, the exponential term in the function is maintained for normalization purposes.

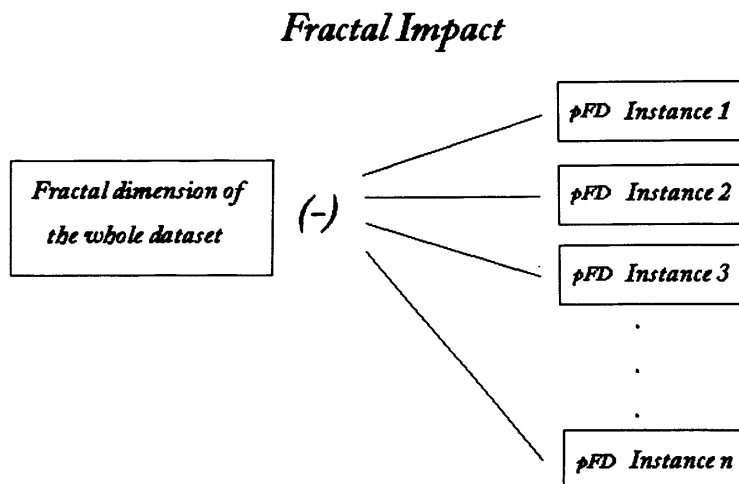


Fig. 4.2: Illustration of the calculation of the FI.

The algorithm proposed has a computational complexity of  $O(M^2N)$  where  $M$  is the number of features and  $N$  is the number of instances. The computational complexity of Eq. (4.4) is reduced from quadratic to linear on the number of instances. The complexity reduction in the algorithm is achieved by modifying the similarity measure  $S$  from the Eq. (4.5). The Euclidean distance  $D$  is replaced by the calculation of the FI on the equation Eq. (4.5). Using the FI eliminates the double loop needed to calculate the entropy  $E$  in the Eq. (4.4) resulting in a simplified entropy equation:

$$E_F = -\sum_{i=1}^I (Sf_i \log Sf_i + (1 - Sf_i) \log(1 - Sf_i)) \quad (4.7)$$

where

$$Sf_i = e^{-\alpha FI_i} \quad (4.8)$$

where FI is used to calculate the distance among instances and is given as follows:

$$FI = FD - pFD_i \quad (4.9)$$

where  $FD$  is the fractal dimension of the whole data set and  $pFD_i$  is the partial fractal dimension of the  $i$ th instances. The  $pFD$  of the  $i$ th instance is calculated removing the  $i$ th instance from the dataset and calculating the  $FD$  of the remaining  $I-1$  instances.

#### 4.6.1 Unsupervised Redundancy

To calculate the redundancy in the features a new ratio  $R_i$  based on the fractal dimension is proposed. Redundancy is quantified as follows:

$$R_i = \frac{\min \Delta pD}{\Delta pD_i} \quad 0 \leq R_i \leq 1 \quad (4.10)$$

where  $i \Delta pD$  is the difference between the  $i$ th partial fractal dimension and the fractal dimension  $D$ . The expression  $\min \Delta pD$  refers to the minimum absolute difference

among all the partial fractal dimensions. When  $i \Delta pD$  tends to be small and approaches  $\min \Delta pD$ , the ratio  $R_i$  takes a value close to its upper extreme, which indicates the redundancy of the  $i$ th attribute. When  $\Delta pD_i$  tends to be large the ratio  $R_i$  approaches zero, which means that the  $i$ th attribute has a low redundancy

$$\max_{x_i \in F} [E(x_i) - R_i(F)] \quad (4.11)$$

where  $E$  is the entropy of each feature. Entropy gives a quantity of the relevance each feature has to describe the possible clusters in the dataset.  $R_i$  is the level of redundancy of the  $i$ th feature calculated using the FD and the  $PFDS$ . The elimination of the redundant features helps to discriminate better the clusters and to reduce the size of the feature subsets.

The pseudo-algorithm of the proposed unsupervised algorithm to select important features is shown next:

$T$  = Original Variable set

$M$  = Total number of features in the data

$f_i$  =  $i$ th feature in the data

**Algorithm** – Unsupervised Fractal-Entropy Dimensionality Reduction (UFEDR)

input: dataset  $A$

output: list of attributes in order of their importance

**Begin**

1- Calculate the FD (fractal dimension) of the whole data

**For**  $i=1:M$  {each feature in the dataset}

2-  $T_i = T - f_i$  {the complete set of features without the  $i$ th feature}

3- Calculate the entropy  $E_F$  of  $T_i$  using (4.5)

4- Compute the partial fractal dimension  $PD$  for each feature;

5- Compute  $R_i$  defined in

6- Execute the proposed function (4.9)

**End**

## 4.7 Experimental Results

The experiments in this chapter were carried out to measure usefulness of the subsets of features found by the UFS algorithm proposed in this chapter. The solution assumes that the goal is to obtain at least the same quality of clustering results with a reduced set of features than the clustering results obtained with the whole set.

The proposed algorithm is compared with more FS algorithms. The first one, called unsupervised statistical entropy, is an FS algorithm proposed in (Dash et al., 1997) that uses the Euclidean distance to measure entropy. The second algorithm, called supervised variance of predictive error, is a FS that selects important features in a supervised way. The SFS algorithm for which the accuracy graph can be seen at the bottom of each dataset shows a comparative reference for the above UFS algorithm algorithms.

Benchmark datasets from the UCI machine learning repository website and a real application dataset for wood defects detection were used to select important features.

The quality of features selected by the three FS algorithms was tested using a hierarchical algorithm called *two step* that finds clusters by dividing and merging instances in the data. A detailed description of the *two-step* algorithm can be found in Appendix D.

The data was partitioned randomly in two sets of instances. The first set contains 80% of the data and is used to train the ANN model. The second partition contains 20% of the data and is used to test the model created. The experiment for each subset of features was repeated twenty times.

<b>Dataset</b>	<b># Instances</b>	<b># Features</b>	<b># Classes</b>
Breast Cancer Wisconsing (original)	699	10	2
WDBC (Diagnostic Breast Cancer)	569	32	2
WPBC (Prognostic Breast Cancer)	198	34	2
Australian Credit approval	690	14	2
Echocardiogram	132	12	2
Ecoli	336	8	8
Statlog (Heart)	270	13	2
Hepatitis	155	19	2
Ionosphere	351	34	2
Iris	150	4	3
Lymphography	148	18	4
Parkinson	194	23	2
Pima Indian Diabetes	768	8	2
Lenses	24	4	3
Glass Identification	214	9	6
Tae	151	5	3
Mammography Masses	830	6	2
SPECTF Heart	267	44	2
Vehicle	846	18	4
Vowe Context	990	10	11
Wine	178	13	3
Zoo	101	16	7
Wood (Real Application)	232	17	13

Table 4.1 Description of benchmark datasets.



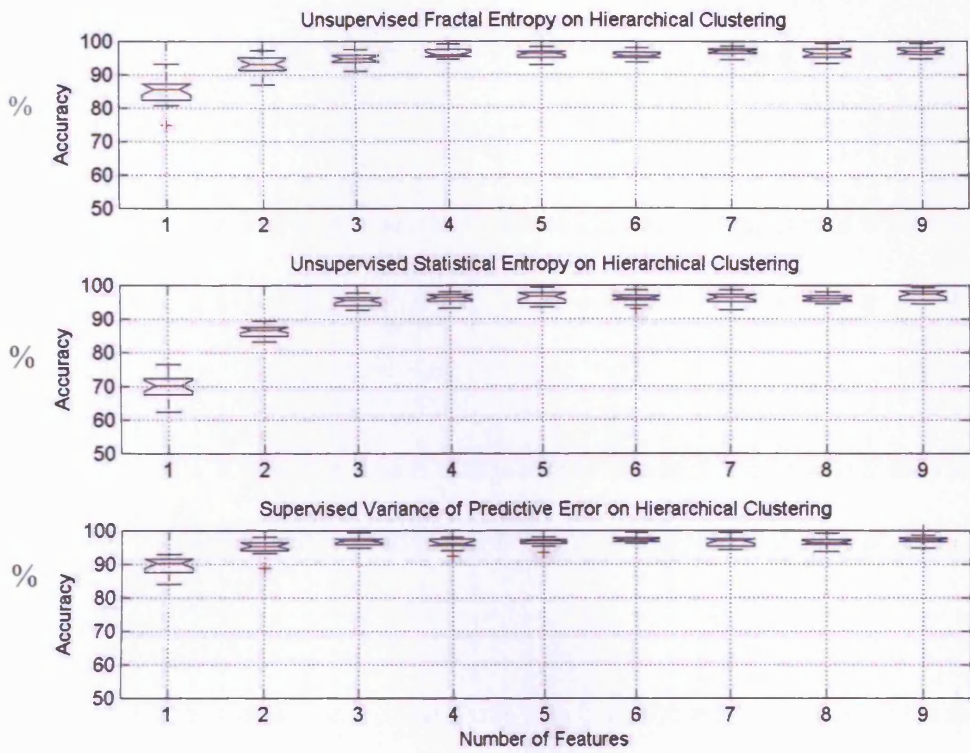


Fig. 4.3: Breast cancer Wisconsin unsupervised and supervised FS comparison.

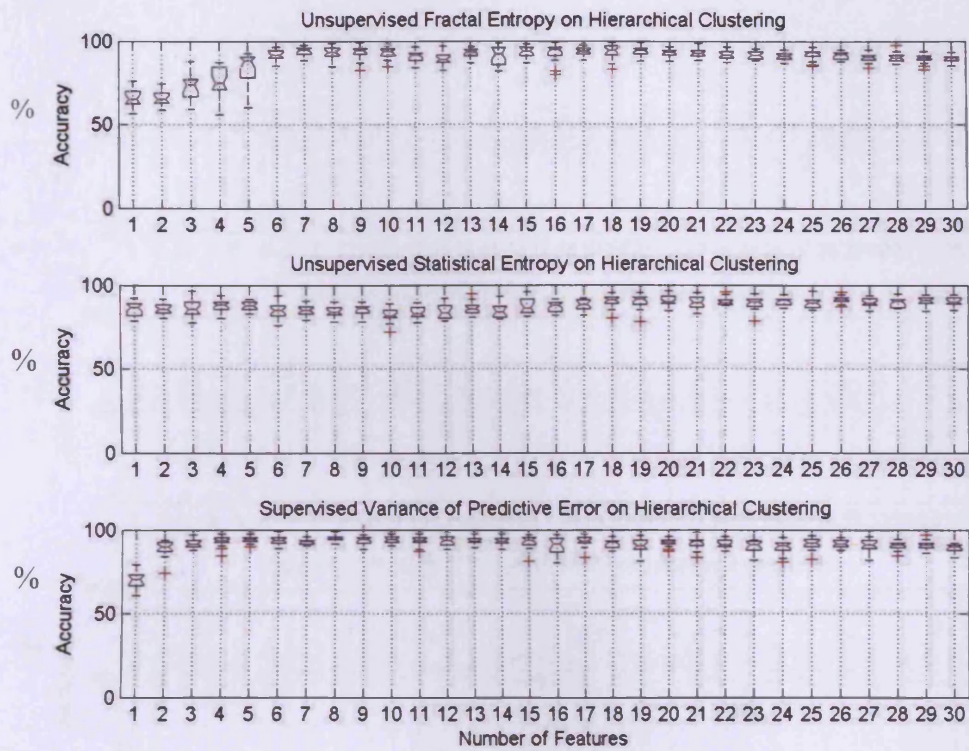


Fig. 4.4: Breast cancer Wisconsin Diagnostic (WDBC) unsupervised and supervised FS comparison.

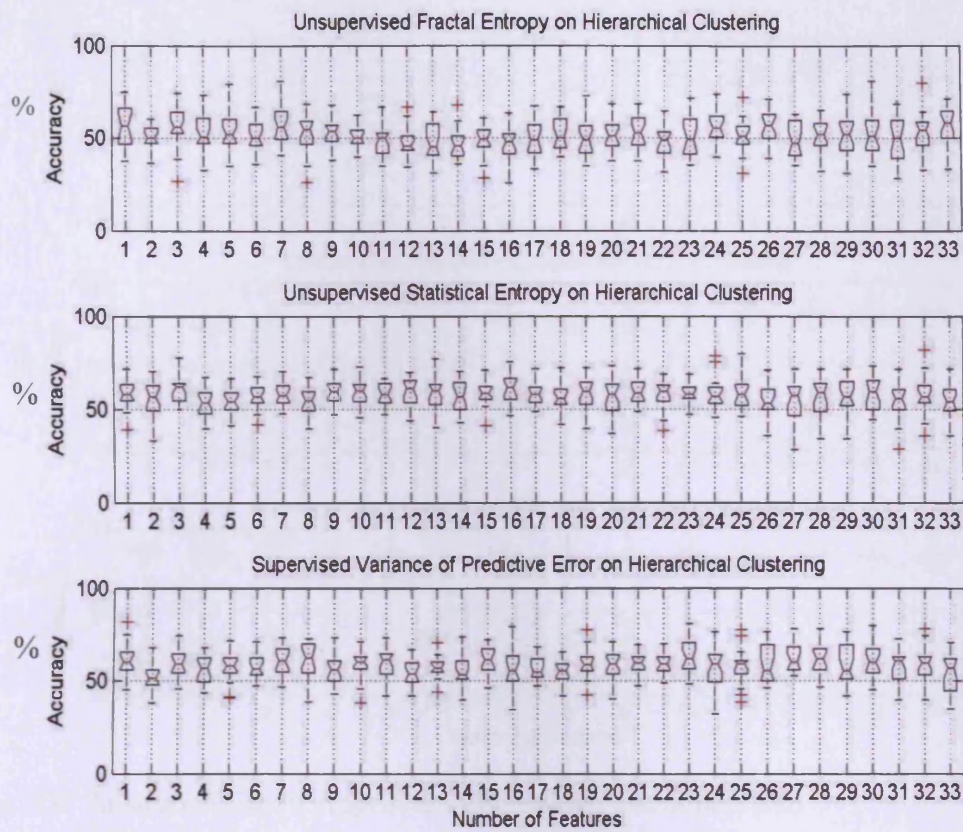


Fig. 4.5: Breast Cancer Wisconsin Prognostic (WPBC) unsupervised and supervised FS comparison.

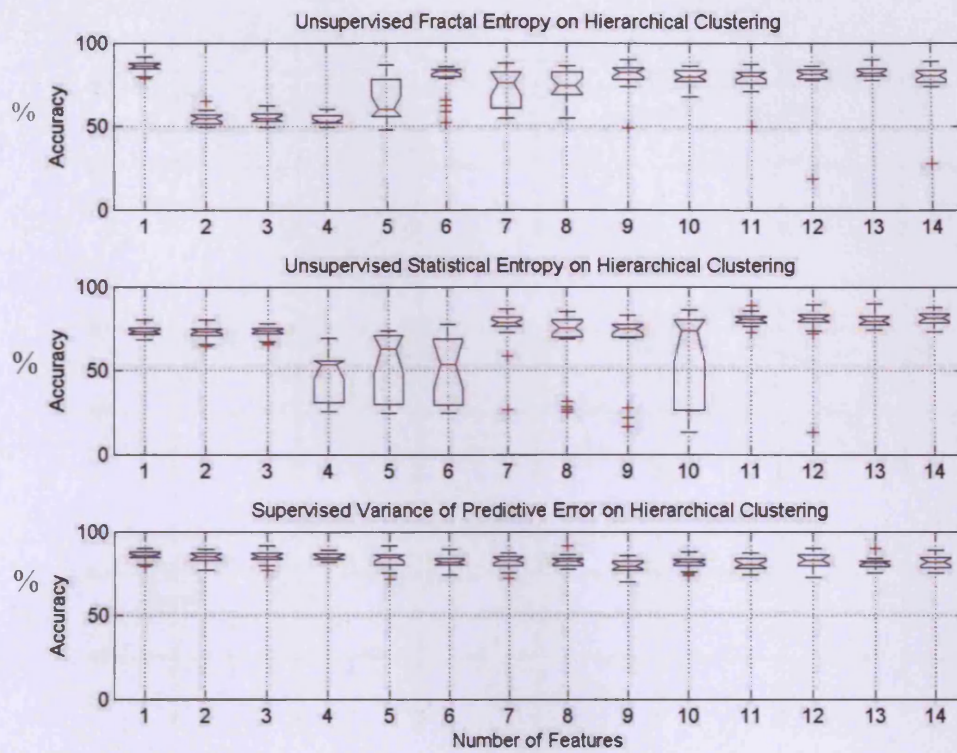


Fig. 4.6: Australian Credit Approval unsupervised and supervised FS comparison.

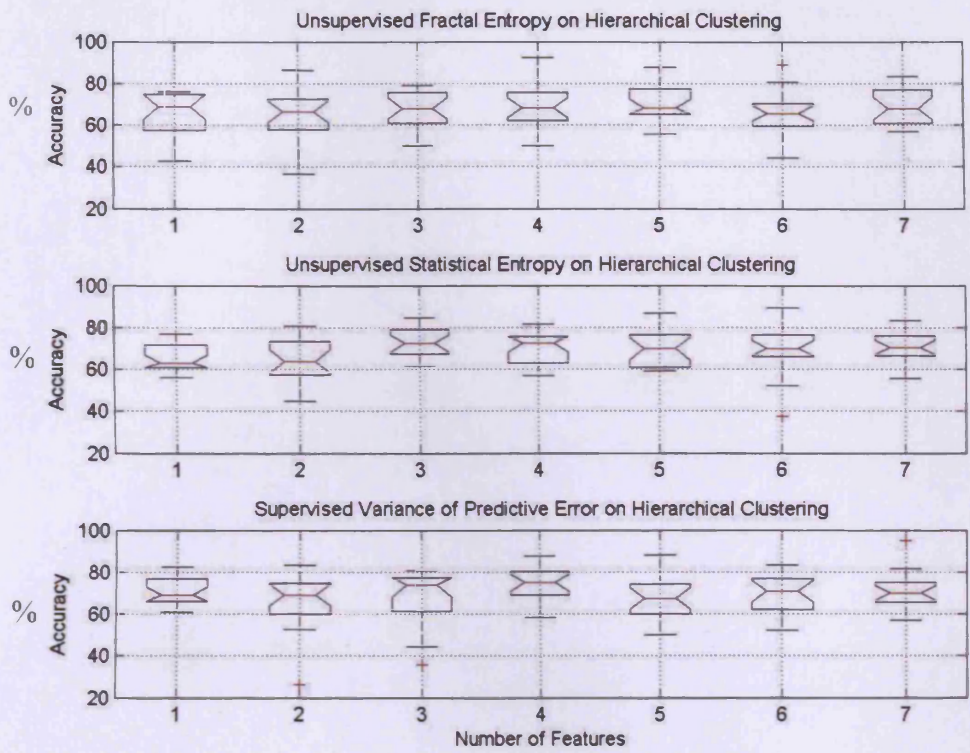


Fig. 4.7: Echocardiogram unsupervised and supervised FS comparison.

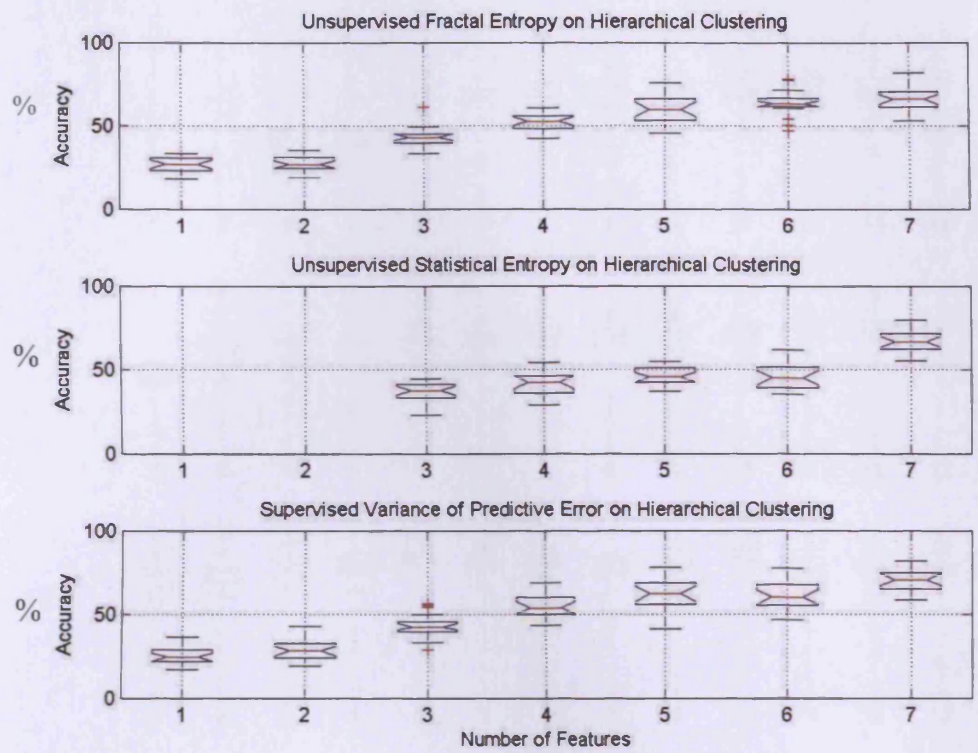


Fig. 4.8: Ecoli unsupervised and supervised FS comparison.

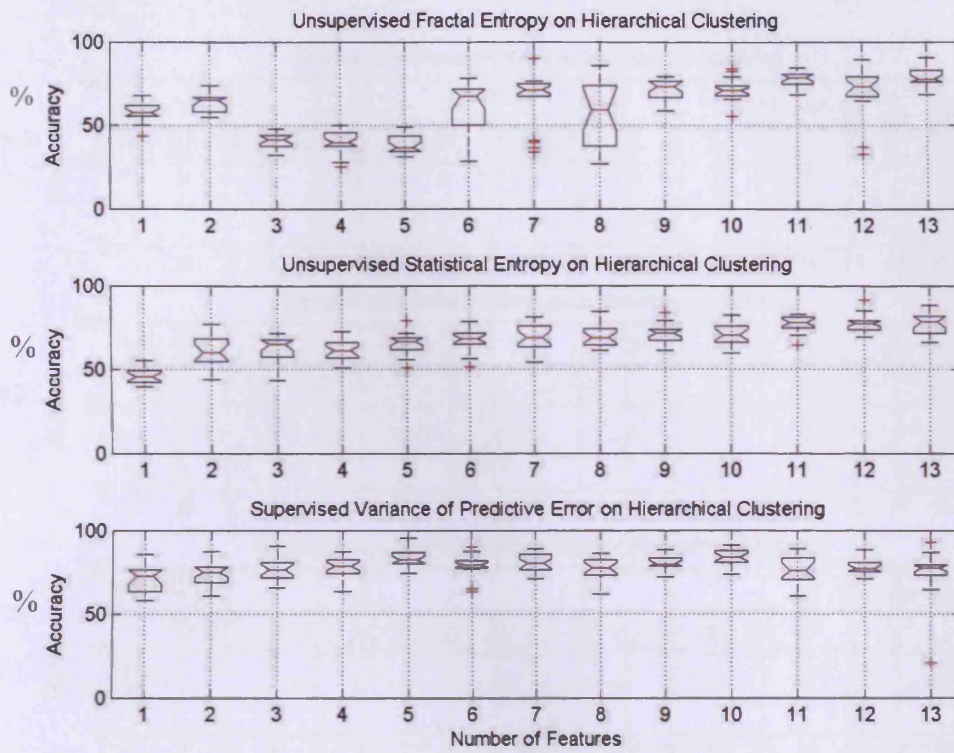


Fig. 4.9: Statlog Heart unsupervised and supervised FS comparison.

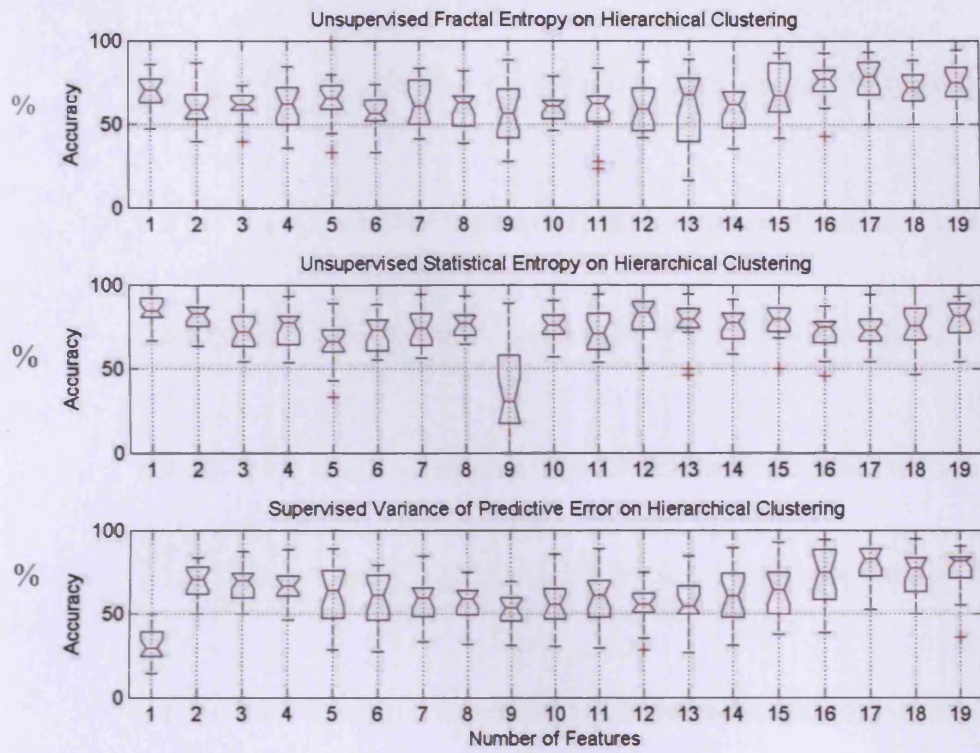


Fig. 4.10: Hepatitis unsupervised and supervised FS comparison.



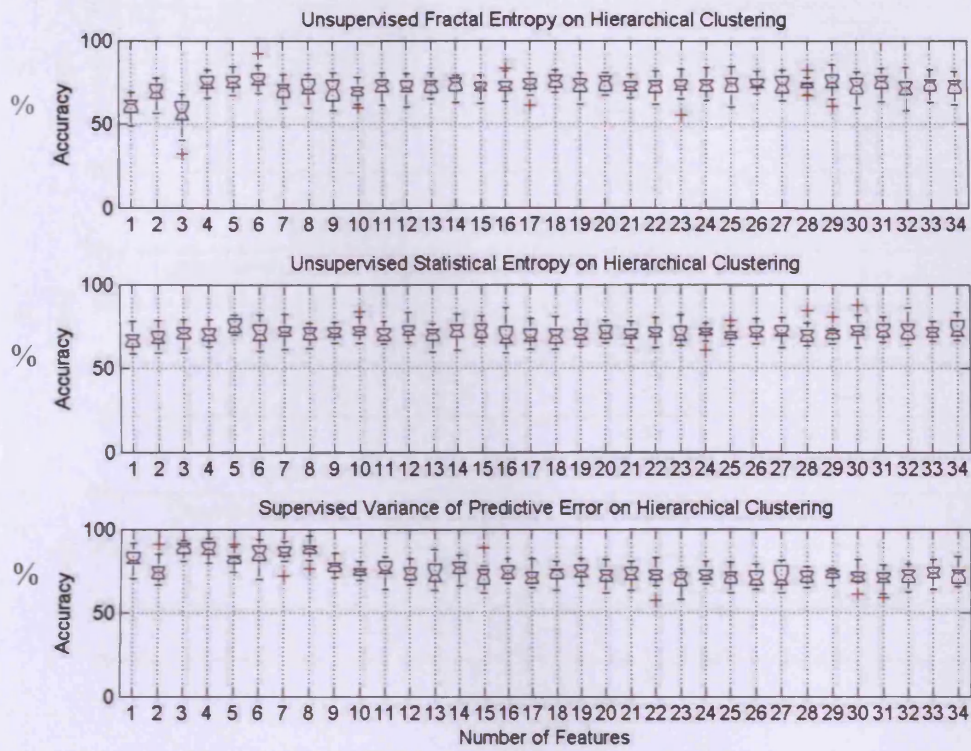


Fig. 4.11: Ionosphere unsupervised and supervised FS comparison.

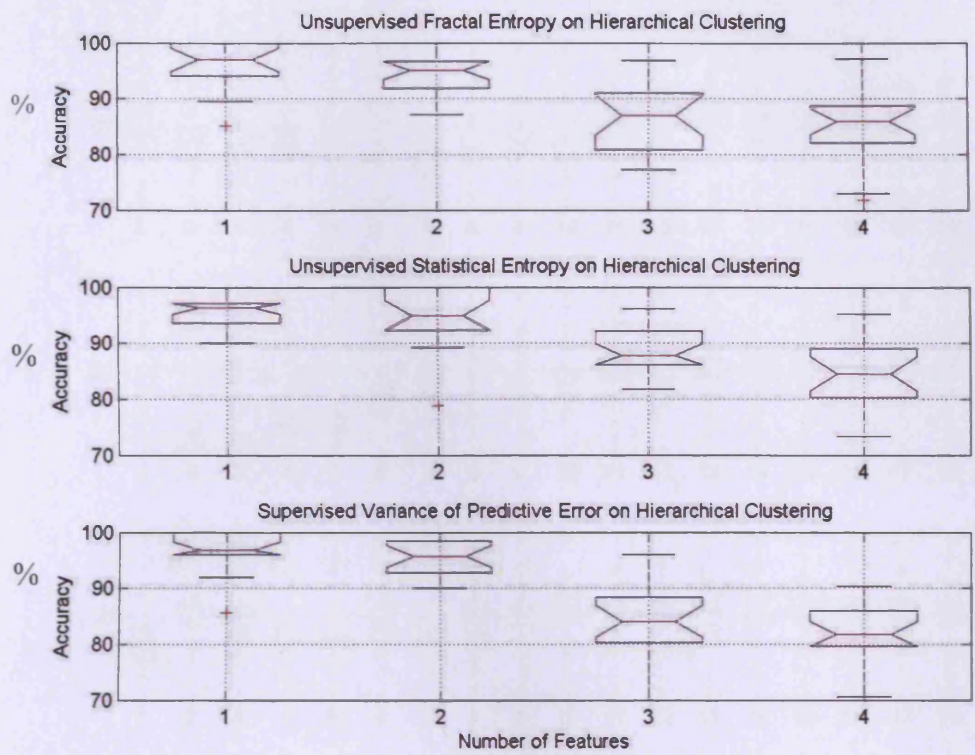


Fig. 4.12: Iris unsupervised and supervised FS comparison.

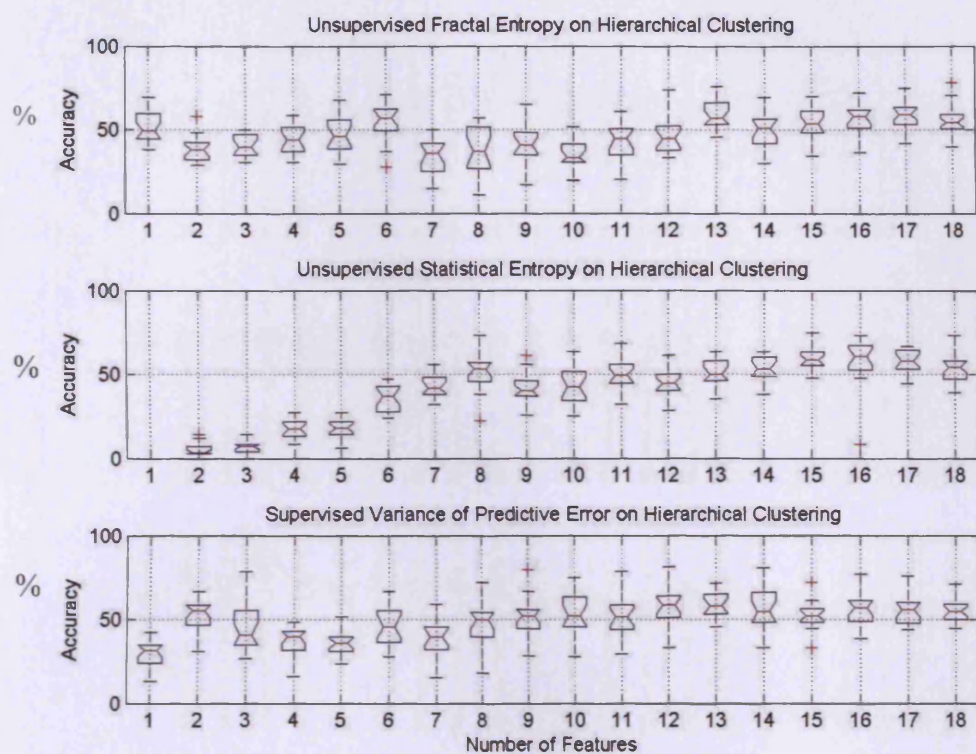


Fig. 4.13: Lymphography unsupervised and supervised FS comparison.

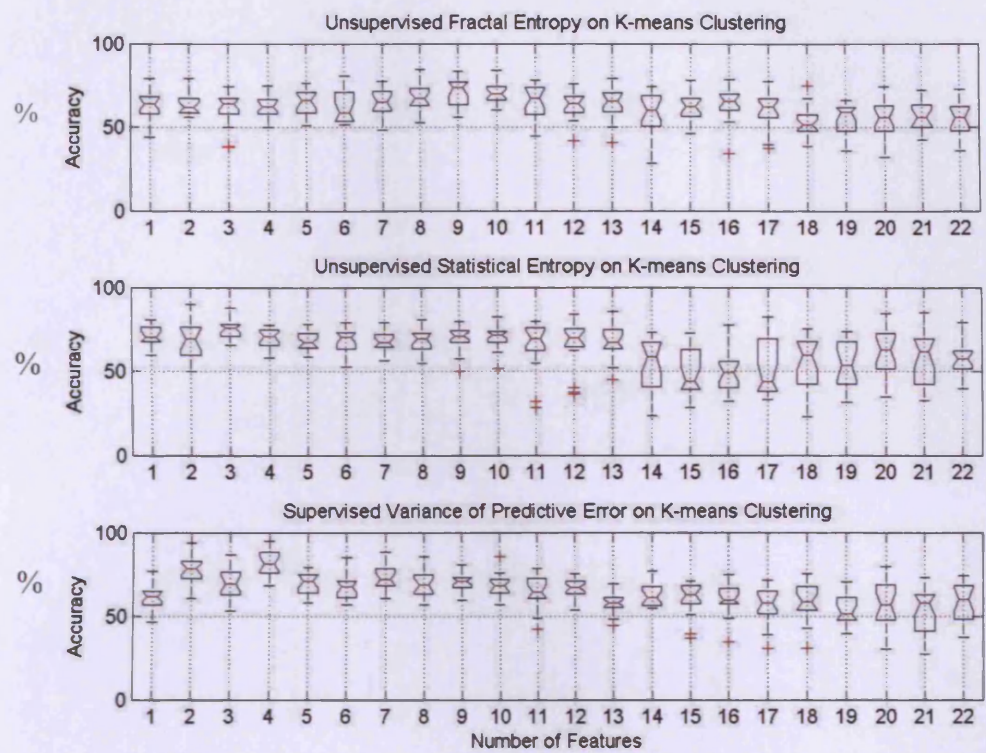


Fig. 4.14: Parkinson (K-means) unsupervised and supervised FS comparison.

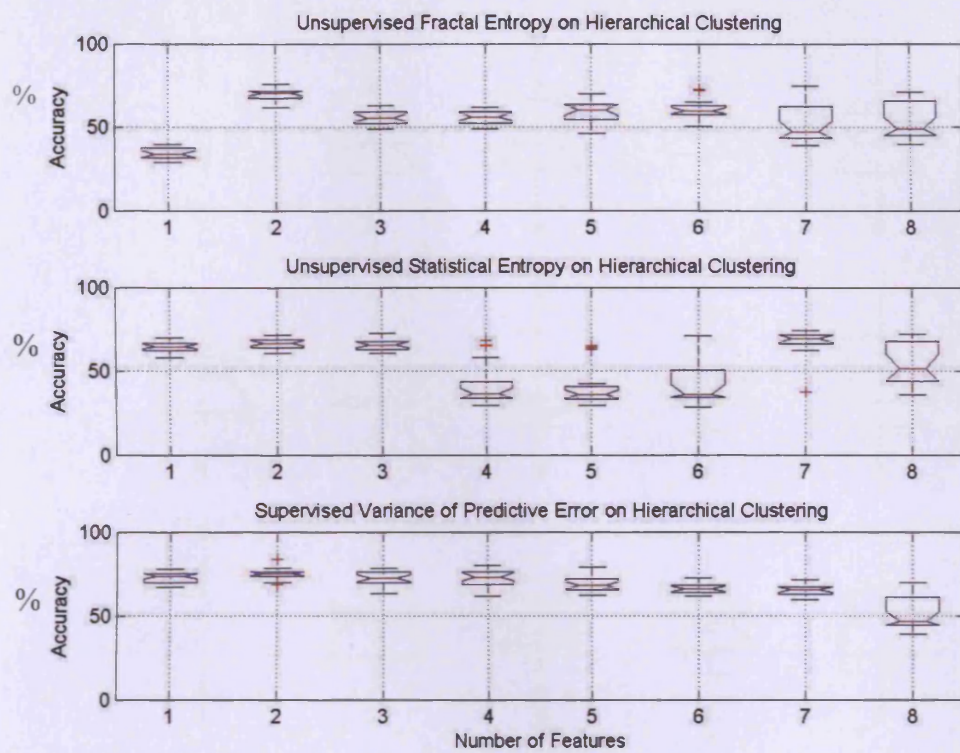


Fig. 4.15: Pima Indian Diabetes unsupervised and supervised FS comparison.

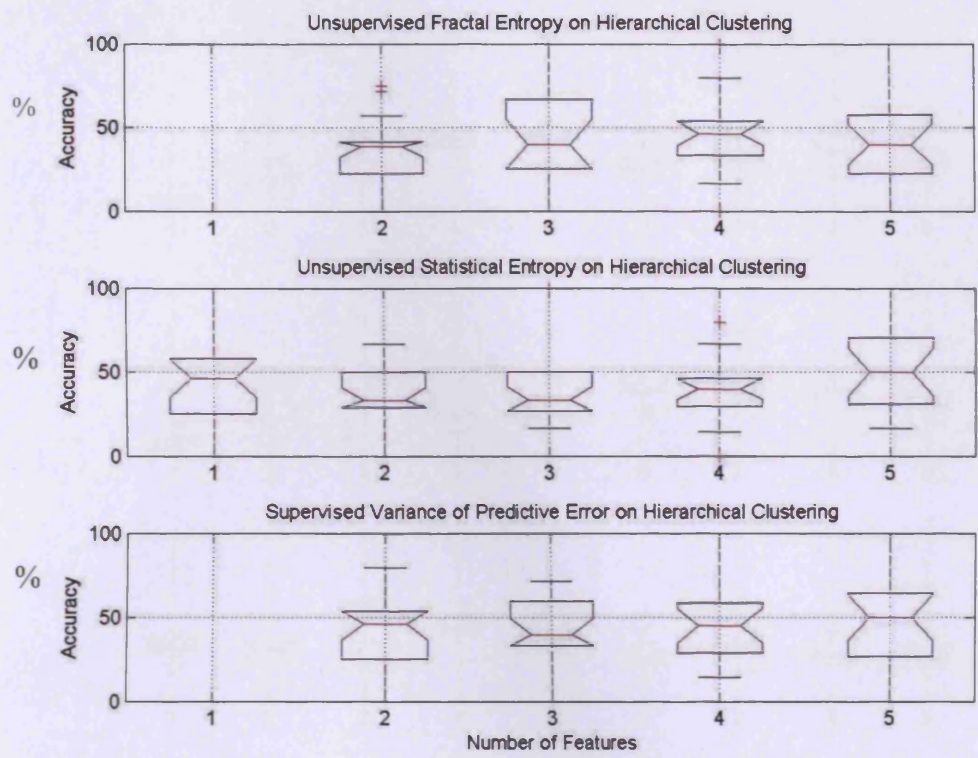


Fig. 4.16: Lenses unsupervised and supervised FS comparison.

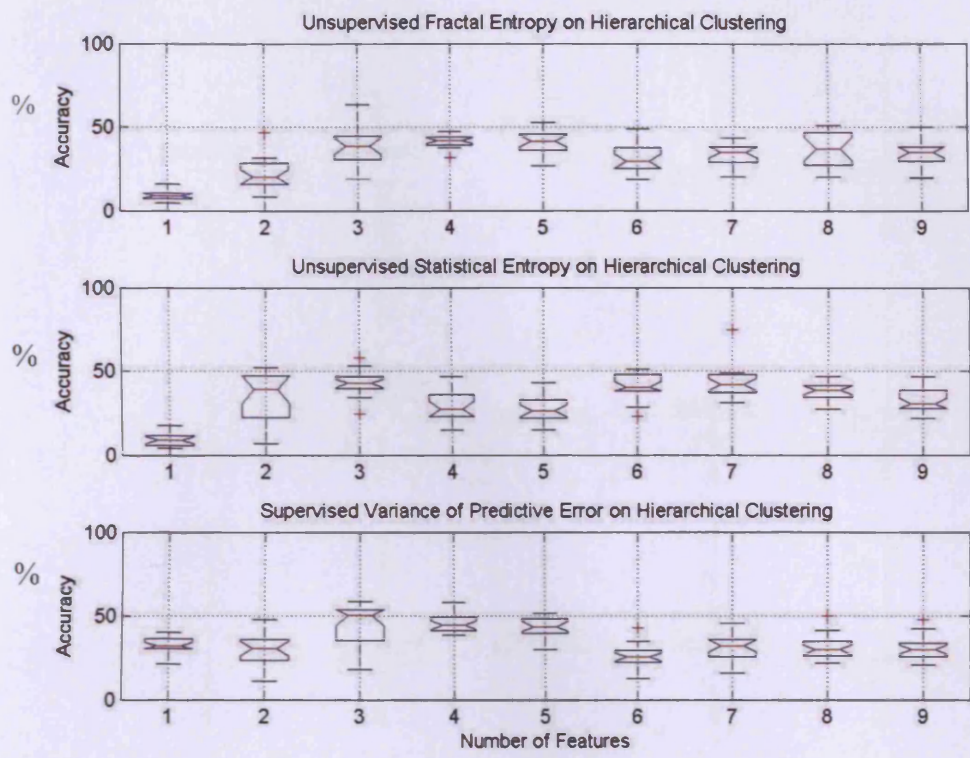


Fig. 4.17: Glass Identification unsupervised and supervised FS comparison.

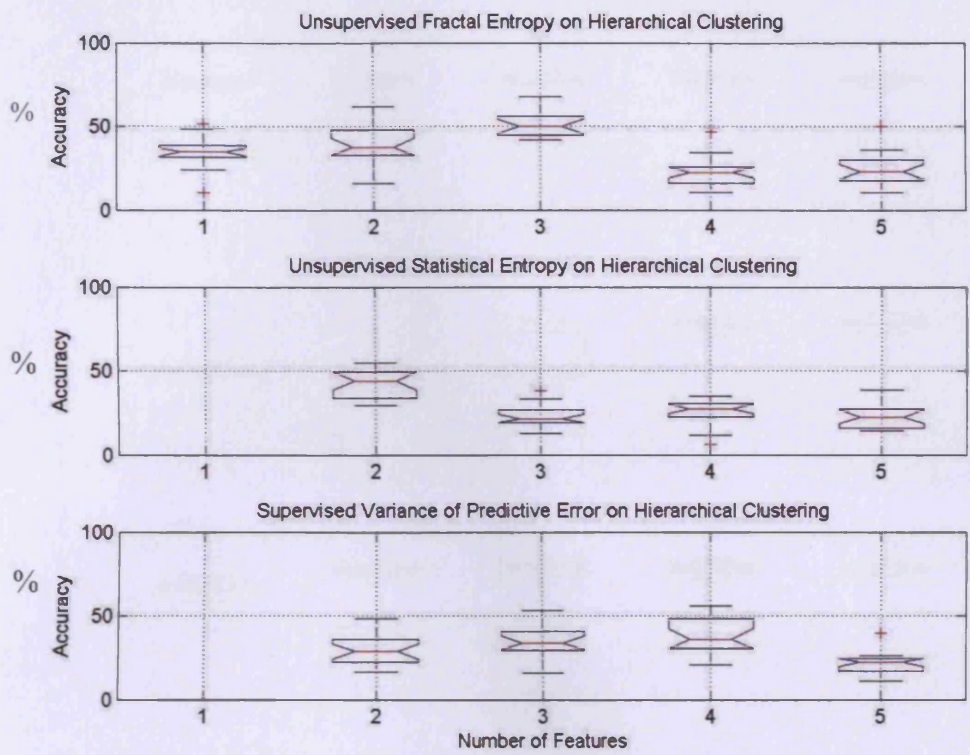


Fig. 4.18: Tae unsupervised and supervised FS comparison.



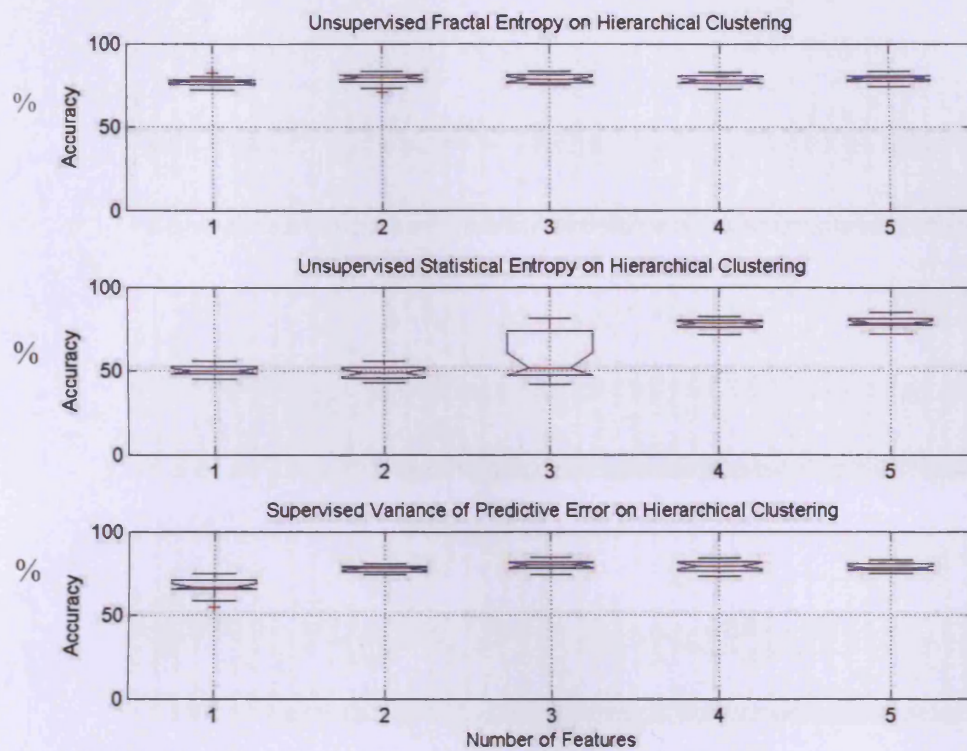


Fig. 4.19: Mammography Classes unsupervised and supervised FS comparison.

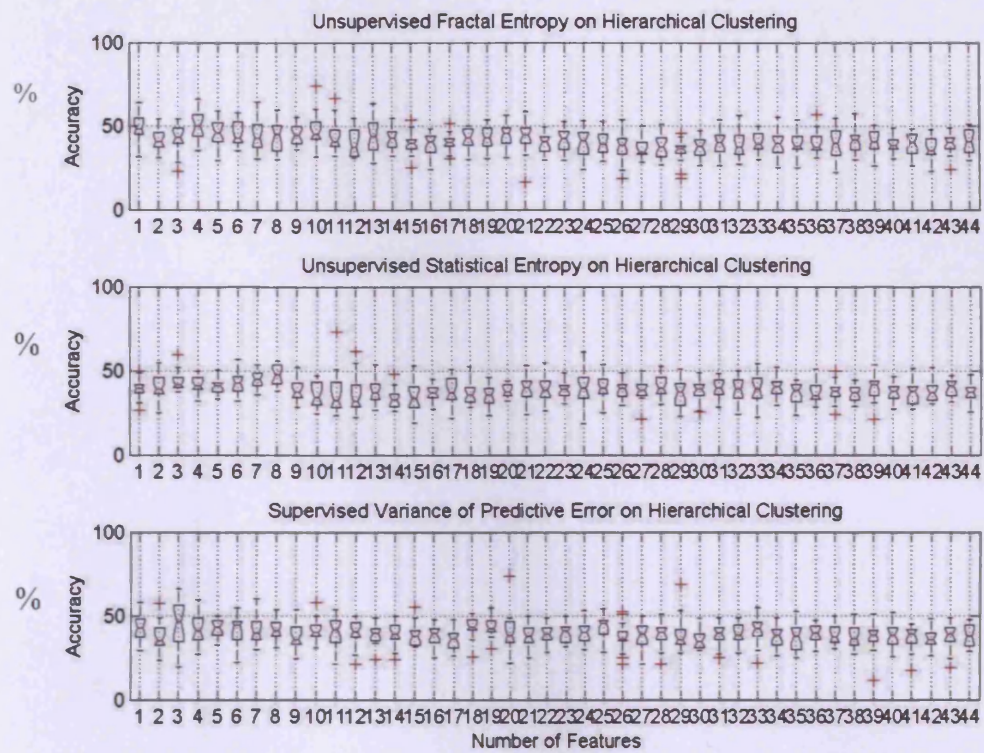


Fig. 4.20: SPECTF Heart unsupervised and supervised FS comparison.

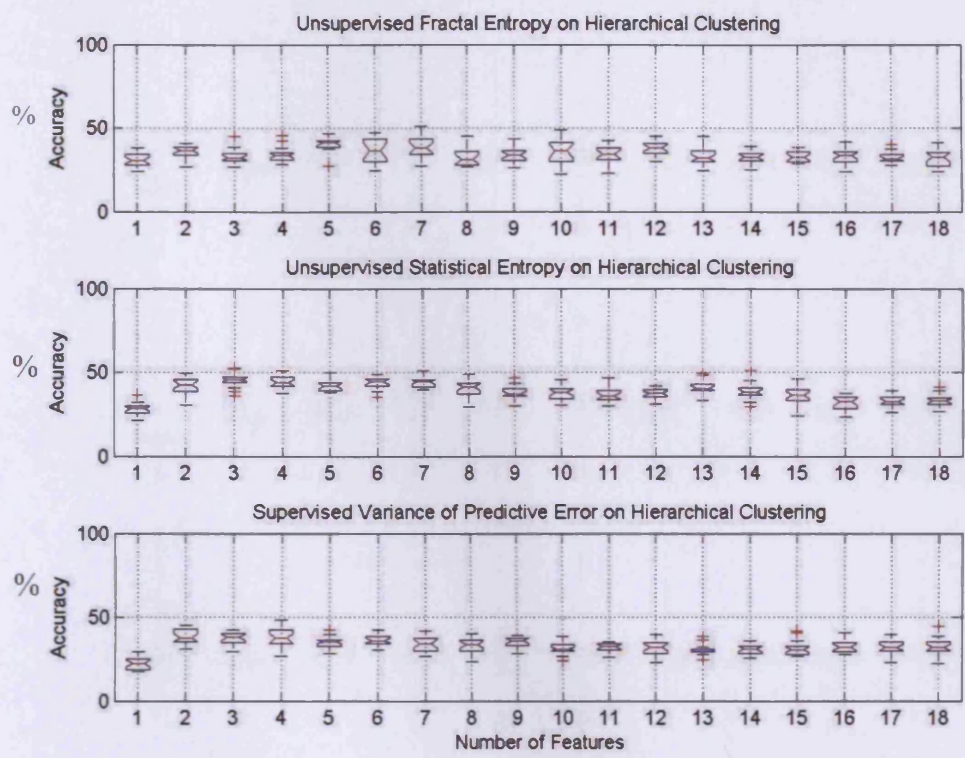


Fig. 4.21: Vehicle unsupervised and supervised FS comparison.

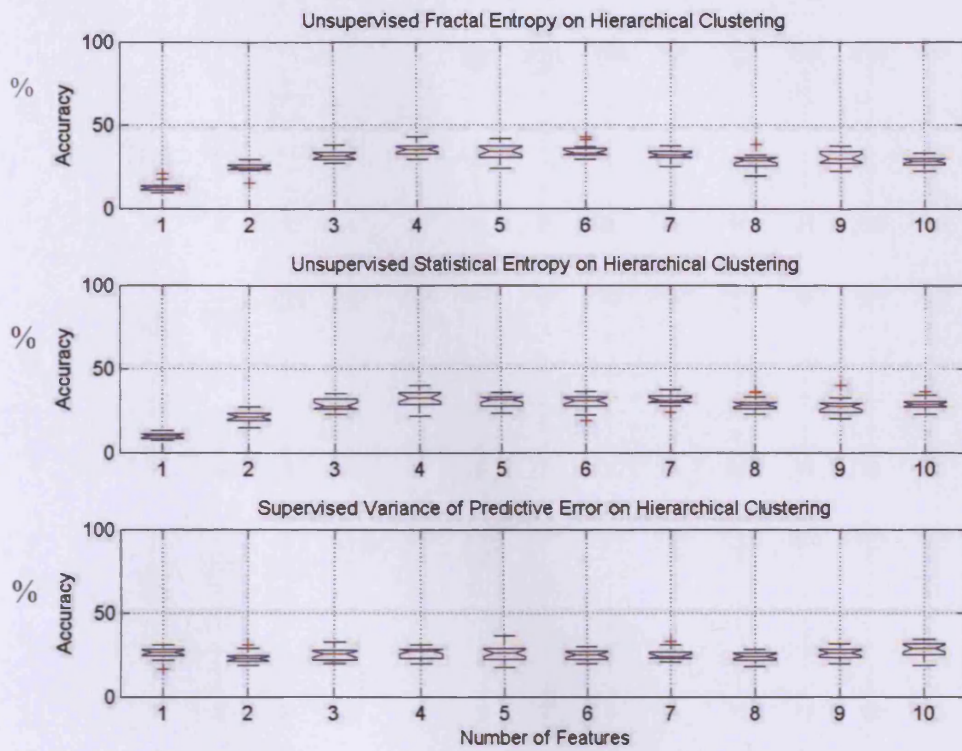


Fig. 4.22: Vowe Context unsupervised and supervised FS comparison.

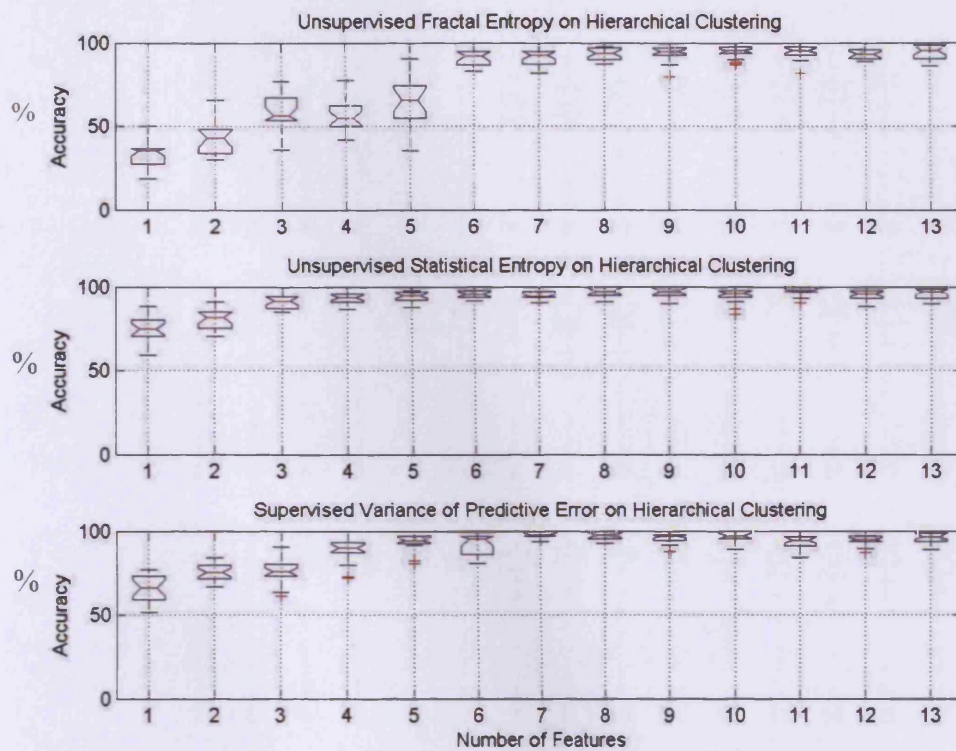


Fig. 4.23: Wine unsupervised and supervised FS comparison.

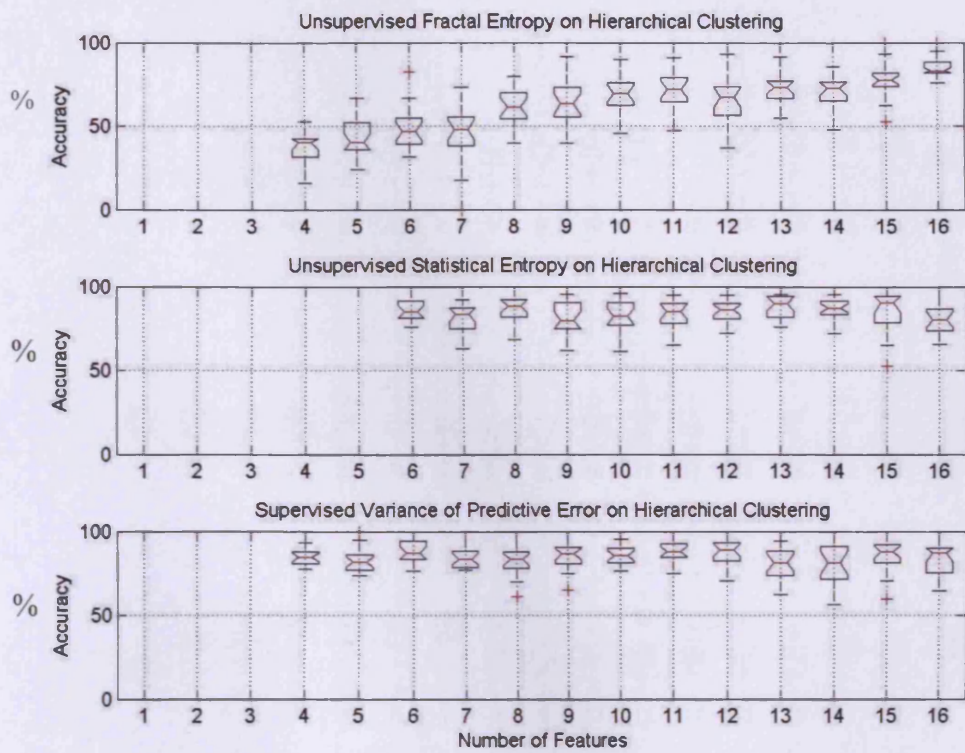


Fig. 4.24: Zoo unsupervised and supervised FS comparison.

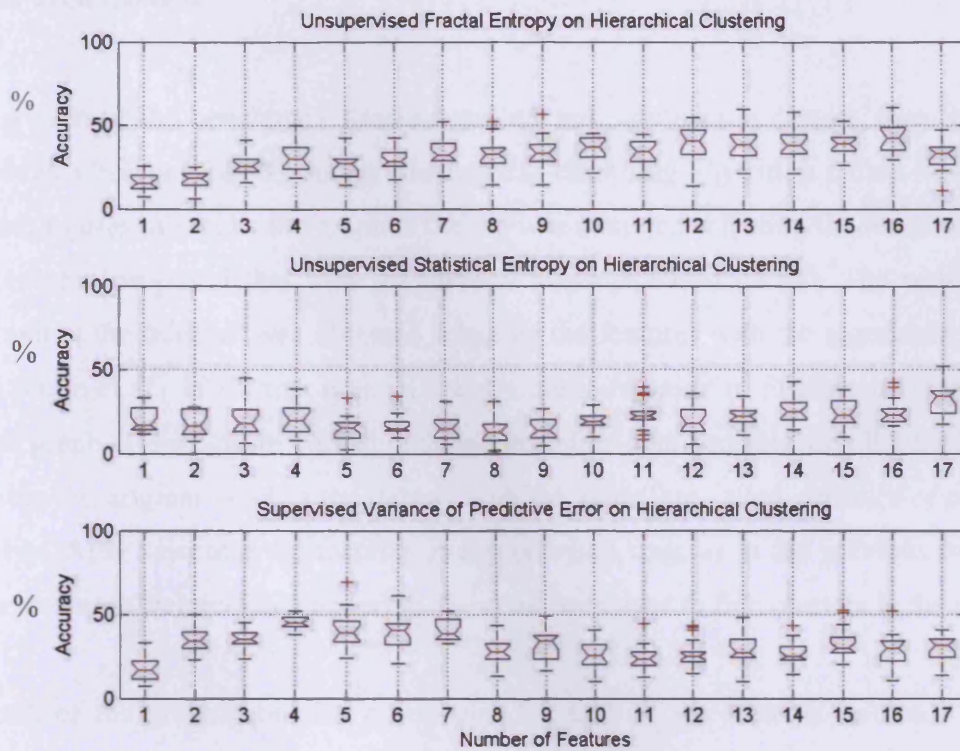


Fig. 4.25: Wood (real application) unsupervised and supervised FS comparison.

## 4.8 Discussion

For each of the benchmark datasets and the real application dataset, three graphs of results were generated using a hierarchical clustering algorithm called *two step*. In each figures of results the graph at the top was obtained selecting the features with the algorithm proposed that uses the entropy measure based on FD. The results in the graph in the middle were obtained selecting the features with the algorithm proposed in (Dash et al., 1997) that uses an entropy measure based in Euclidean Distance. The last graph at the bottom corresponds to the results obtained selecting the best features using the original label in the dataset with the algorithm called *variance of predictive error*. After selecting the features in a supervised way, as in the previous two cases, the *two step* hierarchical clustering algorithm was used to find clusters in the datasets.

Each of the graphs contains a box plot for each of the features selected. The best feature is used alone and then combined with the second best feature, after this, the pair of best features is combined with the third best feature and so on, until the last feature is combined. The quality of the clusters is evaluated using the original label in the dataset. A feature is considered good when it leads to a good clustering i.e., the size of the box is small and with high accuracy. The size of the box gives information of how stable is the set of features over 20 repetitions of the algorithm. As more stable the set of features is, as smaller the box will be. If the set of features gives non-stable results the box will be large. The horizontal line in the box is the sample median, the tops and bottoms of each box are the 25<sup>th</sup> and 75<sup>th</sup> percentiles of the samples, respectively. The distances between the tops and bottoms are the interquartile ranges.

The whiskers are lines extending above and below each box. The whiskers are drawn to show the furthest observations within the whisker length, adjacent values. Samples beyond the whisker length are marked as outliers. The notches give information about the variability of the median between samples. When the notches of two different boxes do not overlap have different medians at the 5% significance level. Comparing box–plot medians is like a visual hypothesis test, analogous to the t-test used for means.



In most of the dataset the UFS algorithm proposed in this chapter manage to decrease the number of features maintaining the quality of the clustering and in some cases improving it.

Comparatively with the UFS algorithm proposed in (Dash et al., 1997) the fractal based algorithm proposed in this chapter found a better set of features for datasets such as Lymphography, WDBC, Echocardiogram, Ecoli, Tae, mammography classes. The fractal algorithm found a better first feature in datasets such as Statlog heart, SPECTF and show more stable results in datasets such as Australian credit approval and Parkinson.

## 4.9 Summary

The proposed algorithm ranks features in terms of importance using the FD. The relevance is measured using a new entropy function that employs the FD. The algorithm ranks useful features when class information is not available.

This filter algorithm is efficient as it calculates the usefulness of the features in a process in which any search technique is involved. A new entropy measure based on the calculation of the FD is proposed in order to detect relevant features. The new fractal-entropy measure is less computationally expensive on the number of instances than the state-of-the-art entropy measure proposed in (Dash et al., 1997). The original entropy function based on Euclidean distance needs two loops to calculate the entropy of the data, while the proposed entropy function based on FD needs only one loop to perform the calculation. A property of the FD is its fully mathematical nature which removes the sequential backward selection step in the proposed algorithm.



## Chapter 5

### Supervised Feature Selection

#### 5.1 Preliminaries

As previously discussed, learning can occur within two contexts: unsupervised (clustering) and supervised (classification). This chapter presents relevance and redundancy problems analyses in FS and introduces a new algorithm to select important features in a supervised context.

FS has an important purpose in machine learning and is usually applied as a data pre-processing step. Its objective is to choose a subset from the original set of features that describes a data set and to prepare it as an input for learning methods. Reducing the dimension (number of features) of a dataset permits to train faster and create simpler learning machines.

The only way to guarantee the best subset of features is to exhaustively search for all possible combinations of the features available. This algorithm is called exhaustive search. Unfortunately with current technology available exhaustive search is not yet feasible in data mining.

In order to achieve an optimal feature subset a large number of search techniques have been proposed over the years. Hill climbing, beam search and genetic algorithms are among the most popular. Search techniques for FS can be used under three different frameworks: filter, wrapper and embedded. The filter approach is the least computational expensive due to its efficient evaluation performance. The efficiency relies on avoiding feedback from learning machines to evaluate the quality of the features and drive the search.

Over years many filter applications for FS have been proposed to measure the relevance of the features within the class label. The relevant features are chosen in

order to build more “intelligent” computational machines, able to discriminate between the different groups in the data. *Relief* is one of the first and most popular FS filter algorithms. “Optimal cell damage” is another classic filter technique based on ANNs. These algorithms measure the quality of each feature base only on its relevance to the label. Other algorithms like the ones proposed in (Peng et al., 2005) and (Liu et al., 2008) analyses the quality of the features based on two criteria, relevance and redundancy. The reason to add a redundancy analysis is because redundant features do not provide any extra individual information to a feature subset. Therefore the elimination of these features could not damage the informative quality of the subset.

The FS framework that joins the relevance and the redundancy criterion is called maximum Relevance and Minimum Redundancy (mRMR) (Peng et al., 2005). This framework defines the best feature as the one which is most relevant, and the least redundant. The relevance of a feature is calculated measuring the correlation between the feature and the target. The redundancy is measured calculating the correlation among the features (Hall, 2000).

In this chapter an algorithm is proposed to minimize the computational burden in the FS approach avoiding heuristics in the process. The proposed FS algorithm ranks features in order of importance, filtering out non-informative ones. The quality of the features is determined by measuring their relevance to the target and their redundancy to the features themselves. The techniques used to measure the relevance and the redundancy use MI and FD respectively. Both techniques are joined, in a new measure, under the framework of mRMR. The proposed measure applies the mRMR framework in a parallel fashion without using any heuristic techniques which burden the FS process.

In order to verify the performance of the proposed algorithm, 29 different benchmark datasets from the UCI machine repository were used. The features in each of the datasets were ranked in terms of importance and used to build ANN classifiers. The quality features were verified, within the class label, by calculating the percentage accuracy generated by the classifiers.

This chapter is organized as follows. Section 5.2 gives an overview of the filter approach, including the MI and the mRMR for SFS. In section 5.3 the algorithm proposed for FS is explained. In section 5.4 the experimental methodology and results are presented. Section 5.5 summarises the chapter.

## **5.2 The Filter Approach**

In (John et al., 1994) the term filter was coined to describe the approach for feature selection that does not use any induction algorithms to evaluate the quality of the features as it uses the data as the only source to measure their quality. This characteristic places filters as the fastest of the three FS approaches. A second advantage of filters is that the features selected should perform optimally in different types of classifiers, although this is not always the case. Filters provide a ranking list with the total number of features included in the dataset. The top feature in the list is considered the most important, and the feature in the bottom is considered the least important.

Some filter algorithms analyse features in an univariate way (Duch et al., 2003). This means that the features are evaluated one by one, not evaluating the performance in combination with any other feature. The univariate method has the disadvantage of making the assumption of feature independence, which may result in lower results compared to the ones obtained using multivariate methods i.e. wrappers and embedded. On the other hand the univariate approach saves computational complexity in not using search techniques as they do not need to explore the feature space in a combinatorial way (Guyon et al., 2006).

### **5.2.1 Filters Approach Based on MI**

MI is a popular technique to characterise general dependence between two variables as it has several advantages over other techniques. MI is a nonparametric and nonlinear technique that makes no assumption about the distribution of the data. MI is able to detect different types of feature dependencies among the features without relying in any transformation of the data. MI has been used successfully to measure

dependency among features (redundancy) and dependencies between features and target (relevance) (Battiti, 1994, Conaire and O'Connor, 2008, Ding and Peng, 2003, Francois et al., 2007, Michel et al., 2008). There are several advantages of using MI instead of correlation techniques. One is that MI is able to detect general dependencies between variables while correlation is only able to detect linear ones. Another important difference between mutual information and correlation function is that MI can be applied to symbolic sequences as well as numeric sequences, but correlation can only be used on numerical sequences (Li, 1990). Nonetheless, when MI is used on tasks characterized by high input dimensionality suffers of inexact results caused by prohibitive computational demands and number of samples. This forces MI to be calculated only by using two-dimensional inputs to obtain accurate results.

MI can be calculated based on Shannon theory (Peng et al., 2005) as it can be used to reduce the uncertainty in a system for example: to calculate relevance, a system in charge of measuring the dependency between features and the target. In this case the system is the classifier which is conformed by the output (class) and the input information vector (features). Mathematically MI is described as follows: uncertainty in the output class can be measured by the entropy

$$H(C) = -\sum_{c=1}^{N_c} P(c) \log P(c) \quad (5.1)$$

where  $P(C)$  is the probabilities for the different classes  $c=1, \dots, N_c$ . The average uncertainty after knowing the feature vector  $F$ , with  $N_f$  components, is the conditional entropy:

$$H(C/F) = -\sum_{f=1}^{N_f} P(f) \left( \sum_{c=1}^{N_c} P(c/F) \log P(c/F) \right) \quad (5.2)$$

where  $P(c/F)$  is the conditional probability for class  $C$  given the input vector  $F$ . In the case where the feature vector is composed of continuous variables, the summatory symbol is replaced by an integral and the probabilities by the corresponding

probability densities. Usually the conditional entropy will be less than or equal to the initial entropy. It is equal if and only if one has independence between features and output class, i.e.; if the joint probability density is a product of the individual densities  $P(c, f) = P(c)P(f)$ . The amount by which the uncertainty is decreased is, by definition, the MI  $I(C;F)$  between  $c$  and  $f$

$$I(C;F) = H(C) - H(C|F) \quad (5.3)$$

The MI is the amount of knowledge provided by the input feature  $F$  which decreases the uncertainty about the class. If the uncertainty is considered within two different events  $(C,F)$ , i.e.,  $H(C;F)$ , usually this is less than the sum of the individual uncertainties  $H(C)$  and  $H(F)$ , the following relation

$$H(C;F) = H(C) + H(F) - I(C;F) \quad (5.4)$$

The combinations of uncertainties  $H(C;F)$  is smaller because of the information that one variable provides about the other one. If the input feature vector has continuous components the MI is defined as follows:

$$I(C;F) = I(F;C) = \sum_c \int P(c, F) \log \frac{P(c, F)}{P(c)P(F)} df \quad (5.5)$$

The MI is directly estimated from the dataset when probability distributions are unknown in advance. The MI is a function of the *joint* probability distribution of the two variables  $c$  and  $F$ , its calculation is a sensitive part of the estimation of the MI. Other methods have been proposed to calculate MI, in (Beirlant et al., 1996, Drugman and Thiran, 2007, Scott, 1992), (Hyvarinen and Oja, 2000, Viola, 1995, Viola et al., 1996) kernel-based and splines are used. In (Zhou et al., 2006) the estimation of the probability density is based on the distribution of the MI in a bayesian framework.

## 5.2.2 Maximal Relevance Minimum Redundancy (mRMR)

FS algorithms should be able to find a good feature subset by eliminating features that provide little or no additional information. The relevance analysis is not sufficient to find a good feature subset, especially in high dimensional datasets where some of the features may be redundant. One may assume that the more features a set has the more discriminative the set is. But having a large number of relevant features has two main disadvantages: the first one is that an excessive number of features will slow down the process of learning. The second disadvantage is that having too many features causes an over-fit in the training data as the learning is excessively constrained due to the amount of features. Therefore a redundancy analysis is needed in order to filter out features that do not provide any extra information (redundant), even if they are relevant to the target (Deisy et al., 2007). The framework that unifies the criterion of relevance and redundancy for FS is called mRMR

$$\max_{X_j \in X - S_{m-1}} [I(X_j; C) - I(X_i - X_j)] \quad (5.6)$$

It has been shown that features selected under the mRMR have more discrimination power. mRMR can integrate different methods for FS such as filters and wrappers to calculate relevance and redundancy. The mRMR is able to find smaller subsets of better features than those using only wrappers or filters alone. Dijck and Hulle, (2007), Ponsa and Lopez, (2007) present an algorithm under the mRMR that uses the MI to find relevant features and a wrapper methodology to perform the redundancy analysis. Ding and Peng, (2003) apply mRMR in five gene expression data sets selecting features with a broader spectrum of characteristics of phenotypes than those obtained through standard ranking methods and the method led to a better classification results. Aunffarth et al, (2008) proposes a Hopfield network selection algorithm based on mRMR, the algorithm is compared with unitary redundancy and relevance filters, obtaining better results. Hejazai and Cai, (2009) proposed an algorithm that finds relevant and non redundant features for water resources systems based on Venn diagrams.

The overall conclusion of the previous works is that a good solution contemplates ignoring redundant features to increase the general quality of the dataset. Subsets that include highly relevant and low redundant features are part of a good dataset representation.

### **5.2.3 Supervised Redundancy**

Since redundancy in features is measured within the features only, the target is not required in the measure procedure. Taking into account the previous, the ratio proposed in chapter four which measures the redundancy among features, can be used in a supervised task. The redundancy in the features is detected by measuring changes in the FD when a feature is eliminated from the original set of features. A feature which does not affect the FD of the whole data is considered to be redundant.

In order to detect important features the fractal ratio is connected to the MI measure to create a new algorithm under the mRMR framework. The proposed algorithm is explained next.

## **5.3 Algorithm Proposed**

MI has been proposed for feature selection in a wide range of applications. The basic practice of this method is to simply select the top-ranked features with higher MI Chow and Huang, (2005). The deficiency of this method is that the features selected as the most informative ones can be correlated to each other.

The aim of the algorithm proposed in this work is to select a subset with the most relevant and less redundant features in order to obtain stable and high accuracy results in supervised learning tasks. To achieve this goal, different frameworks have been proposed in the literature already (Deisy et al., 2007, Yu and Liu, 2004). The sequential analysis structure of these algorithms has two main steps as shown in Fig. 5.1. The first step is carried out to obtain a subset of relevant features from the original dataset. The second step is to measure the level of redundancy on the selected subset of features to measure their level of redundancy.



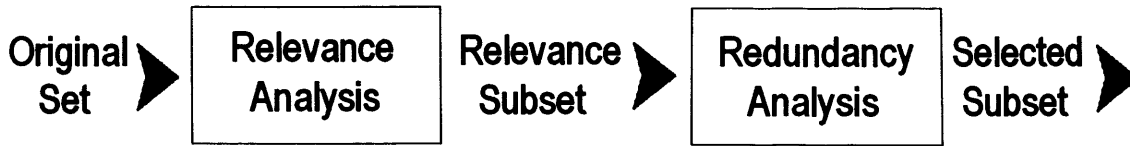


Fig. 5.1: Traditional framework of feature selection.

This sequential feature selection approach has two disadvantages. The first is that redundancy analysis is not applied over all of the features in the dataset, but only in a selected subset of relevant features. As a consequence, attributes of low redundancy that can be useful to describe the dataset may be removed because they are only mildly correlated to the target. The second disadvantage is the necessity of setting the number of relevant features burdening the process after the relevance analysis.

To overcome these disadvantages, a filter algorithm based on a new parallel feature analysis framework is proposed. This algorithm simultaneously evaluates the relevance and redundancy of the features, and balances the feature selection criterion accordingly.

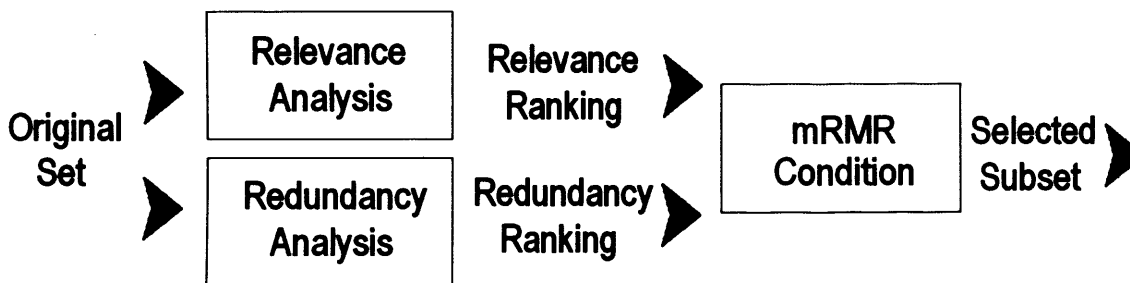


Fig. 5.2: Proposed framework of feature selection.

The first step is to perform the relevance analysis to the original set of features using MI. The MI index of every feature and the target is calculated using Eq. 5.5. The feature which obtains the highest value is ranked top in terms of relevance.

The second step is to quantify the level of redundancy of the whole dataset using FD analysis. This is done by first calculating the FD of the whole dataset and then the  $N$  PFDs. The  $i$ th PFD is calculated eliminating the  $i$ th attribute and computing the fractal dimension  $D_i$  of the remaining  $N-1$  features.

To calculate the redundancy in the features a new ratio  $R_i$  based on the fractal dimension is proposed. Redundancy is quantified as follows:

$$R_i = \frac{\min \Delta pD}{\Delta pD_i} \quad 0 \leq R_i \leq 1 \quad (5.9)$$

where  $i \Delta pD$  is the difference between the  $i$ th partial fractal dimension and the fractal dimension  $D$ . The expression  $\min \Delta pD$  refers to the minimum absolute difference among all the partial fractal dimensions. When  $i \Delta pD$  tends to be small and approaches  $\min \Delta pD$ , the ratio  $R_i$  takes a value close to its upper extreme, which indicates the redundancy of the  $i$ th attribute. When  $\Delta pD_i$  tends to be large the ratio  $R_i$  approaches zero, which means that the  $i$ th attribute has low redundancy. The final step is to verify the following condition:

$$\max_{x_i \in F} [I(x_i, c) - R_i(F)] \quad (5.10)$$

where  $I(x_i, c)$  is the dependency of the  $i$ th feature from the target  $c$  calculated using the MI.  $R_i(F)$  is the level of redundancy of the  $i$ th feature calculated using the FD and the  $PFDs$ . This function is developed by mimicking the mRMR criteria using a different condition for redundancy analysis that avoids the use of search techniques.

The objective of the function defined in Eq. 5.10 is to optimally set a measure of usefulness for the features in the dataset. The full mathematical nature of the MI and the FD allows a pure filter analysis and avoids evaluations of feature subsets using heuristic search strategies. The computational complexity of Eq. 5.10 is linear for the number of instances and quadratic for the number of features.

### 5.3.1 Algorithm Implementation

The algorithm to calculate the fractal dimension  $D$  was implemented in C++ using linked lists instead of arrays to improve memory performance, and connected to MATLAB through MEX files for visualization purposes. The MI computation was

performed using the algorithm proposed in (Peng et al., 2005), using a self-contained and cross-platform package available at Mathworks (MATLAB). The proposed feature selection algorithm using MI and FD with mRMR criteria is shown in Fig. 5.2.

**Algorithm** – MI and FD to Maximise Relevance and Minimize Redundancy

input: dataset  $A$

output: list of attributes in order of their importance

**Begin**

1- Compute the fractal dimension  $D$  of the whole dataset;

**For** each feature in the dataset

2- Compute the partial fractal dimension  $PD$  for each feature;

3- Compute  $R_i$  defined in (5.9)

4- Compute the MI between each feature and the target.

5- Execute the proposed mRMR condition.

**end**

Filter methods usually ranks the features according to a pre-defined criterion of desirability. The decision on how many features to remove can be taken by using either a threshold of importance, or by evaluating their incremental effectiveness using a learning machine.

## 5.4 Experimental Results

The experiments in this chapter were carried out to test a new algorithm based on the mRMR framework. The algorithm proposed in this chapter ranks features in terms of importance. The quality of the features selected with the proposed algorithm was compared with the one produced by two different state-of-the-art algorithms.

The first FS algorithm called variance of predictive error calculates the importance of the features taking into account not only the relevance of the features within the target, but also considering the interactions and correlations among features. This algorithm relates to the importance of each feature in making a prediction using an ANN classifier. This method leaves one feature at a time, and observes the performance of the remaining model i.e. the method just relates to the importance of each feature in making the prediction. The importance of the feature is calculated from the test partition and also takes into account interactions among features, thus it

might be found that only one of two predictors is set as important if both duplicate much of the same information. For a more detailed reference of the variable of importance approach see Appendix B.

The second FS algorithm selects the features based on strength of its relationship to the specified target. This algorithm uses the *Pearson chi-square* test to measure the independence of the target on each feature in the datasets. The importance of each variable is calculated as  $(1-p)$ , where  $p$  is the  $p$  value of the appropriate statistical test of association between the candidate predictor and the target. For a more detailed reference of this algorithm see Appendix B.

Different ANN models were created to assess the incremental performance of features ranked. The ANN's were created using a different number of features, from one to the total number of features in the datasets. The features are ranked in order of importance determined by each of the algorithms.

The ANN's, used to compare the different feature sets, use the Multilayer Perceptron (MLP) network model. All networks have only one hidden layer which contains a maximum of  $[3, (n_i + n_o) / 20]$  neurons, where  $n_i$  is the number of input neurons and  $n_o$  is the number of output neurons. The network is trained using the back propagation method described in Appendix A.

The data was partitioned randomly in two sets of instances. The first set contains 80% of the data and is used to train the ANN model. The second partition contains 20% of the data and is used to test the model created. The experiment for each subset of features was repeated ten times.

In order to show variability of accuracy of the feature subsets selected box plots graphs are used. The graphs for each of the datasets produce  $N$  boxes and whiskers for each of the  $N$  models created using 1 to  $N$  number of features. The box is limited with lines at the lower quartile, medium and upper quartile values. Whiskers extend from each end of the box to the most extreme data values within 1.5 times the interquartile

range sample from the ends of the box. Outliers are data within values beyond the end of the whiskers and are displayed with a red “+” sign.

Notches display the variability of the median between samples. The width of a notch is computed so that box plots whose notches do not overlap have different medians at the 5% significance level. The significance level is based on a normal distribution assumption, but comparisons of medians are reasonably robust for other distributions. Comparing box plot medians is like a visual hypothesis test, analogous to the t-test used for means. The software used to produce the box plots was MATLAB.

The twenty benchmark datasets were all taken from the UCI machine learning repository website. These datasets were chosen according to their variability of classes and the different levels of overlapping between them. The class overlapping of each dataset can be seen from the graphs presented on Appendix C, where the Principal Component Analysis projection of every benchmark dataset used is shown.

<b>Datasets</b>	<b># Instances</b>	<b># Features</b>	<b># Classes</b>
Credit approval	690	14	2
Echocardiogram	107	7	2
Ecoli	336	7	8
Heart	13	270	2
Hepatitis	19	80	2
Image Satellite (Statlog version)	6435	36	5
Image Segmentation	2310	19	7
Letter Recognition	20000	16	26
Ionosphere	351	34	2
Iris	150	4	3
Lenses	24	5	3
Mammography masses	830	5	2
Parkinson	194	22	2
Shuttle (Statlog)	14500	9	7

Sonar	208	60	2
Spambase	4601	57	2
Vehicle	846	18	4
Vowe Context	990	10	10
WDBC (Breast Cancer Diagnostic)	569	30	2
Wine	178	13	3
SPECTF Heart	267	44	2
Zoo	101	16	7
Lymphography	148	18	4
Glass Identification	214	9	6
Tae	151	5	3
Breast Cancer Prognostic (WPBC)			
Pima Indian Diabetes	768	8	2
Breast Cancer Wisconsin (original)	699	10	2
Wood (Real application )	232	17	13

Table 5.1 Description of benchmark datasets

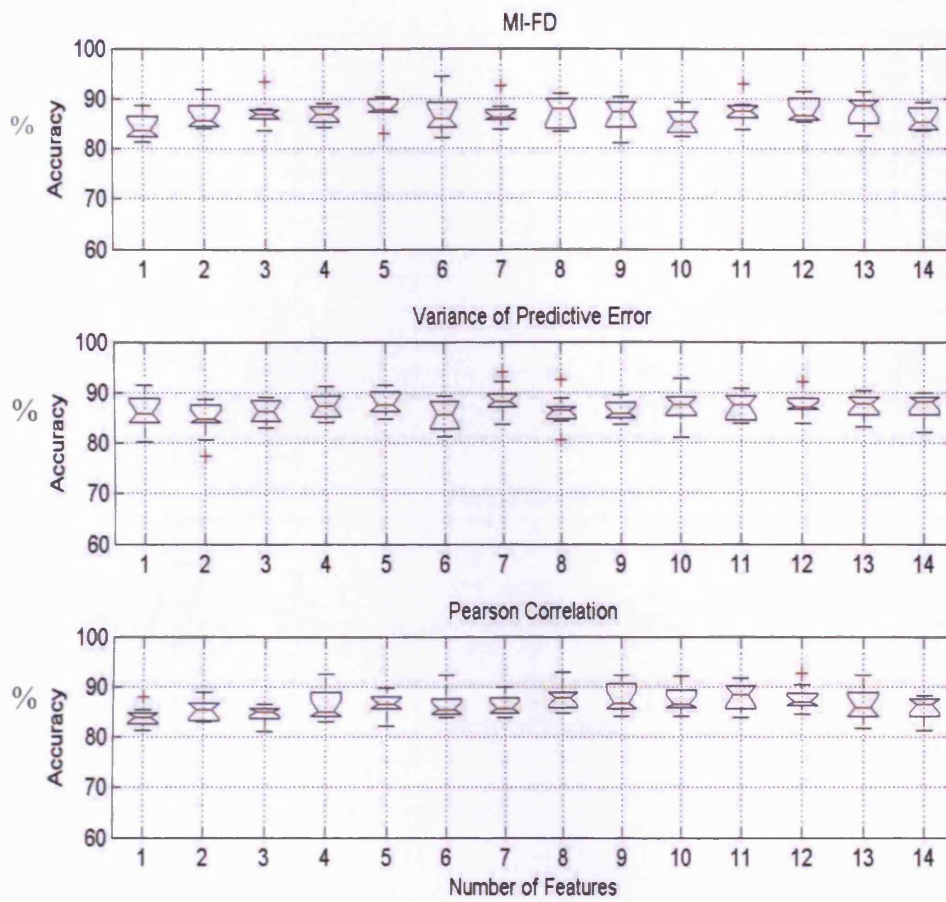


Fig. 5.3: Australian Credit approval SFS comparison.

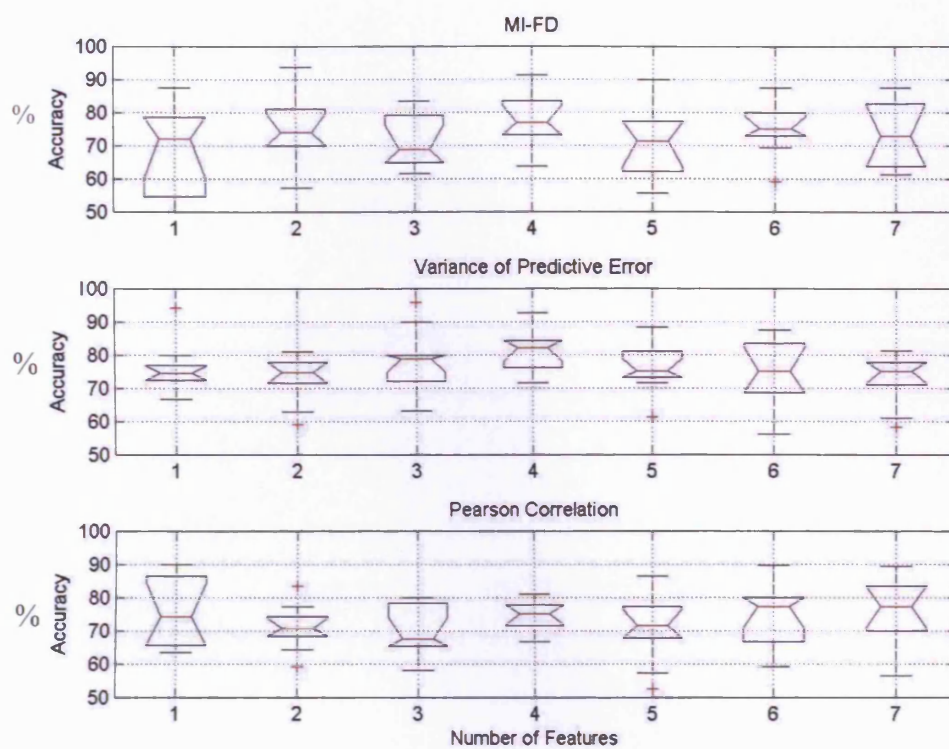


Fig. 5.4: Echocardiogram SFS comparison.



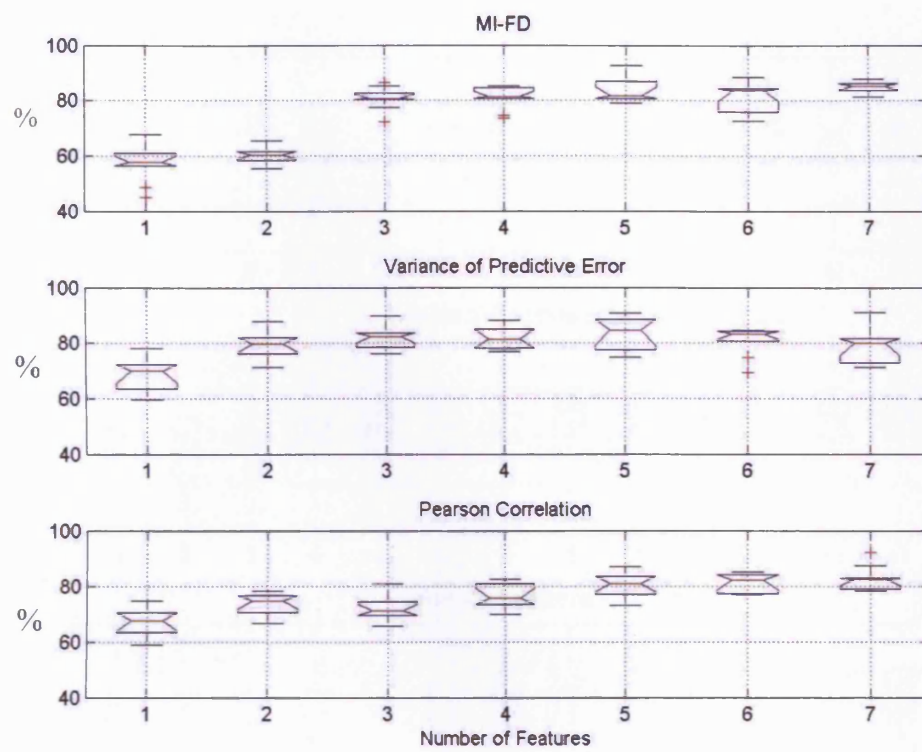


Fig. 5.5: Ecoli SFS comparison.

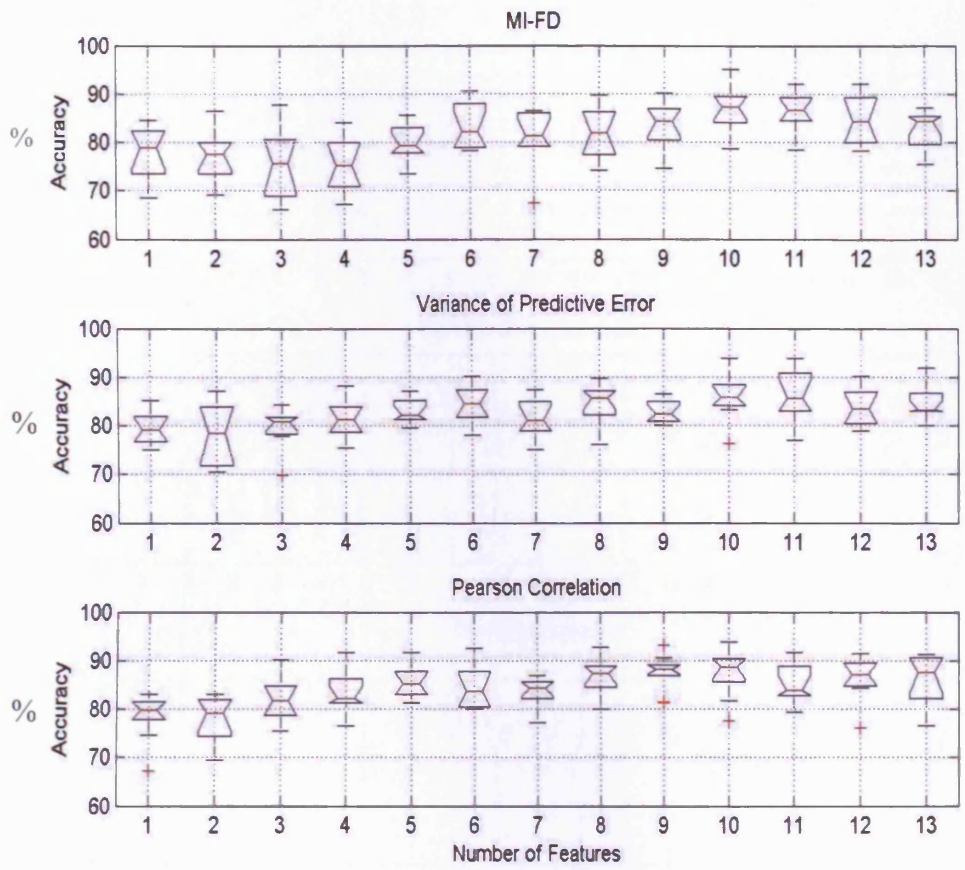


Fig. 5.6: Heart SFS comparison.

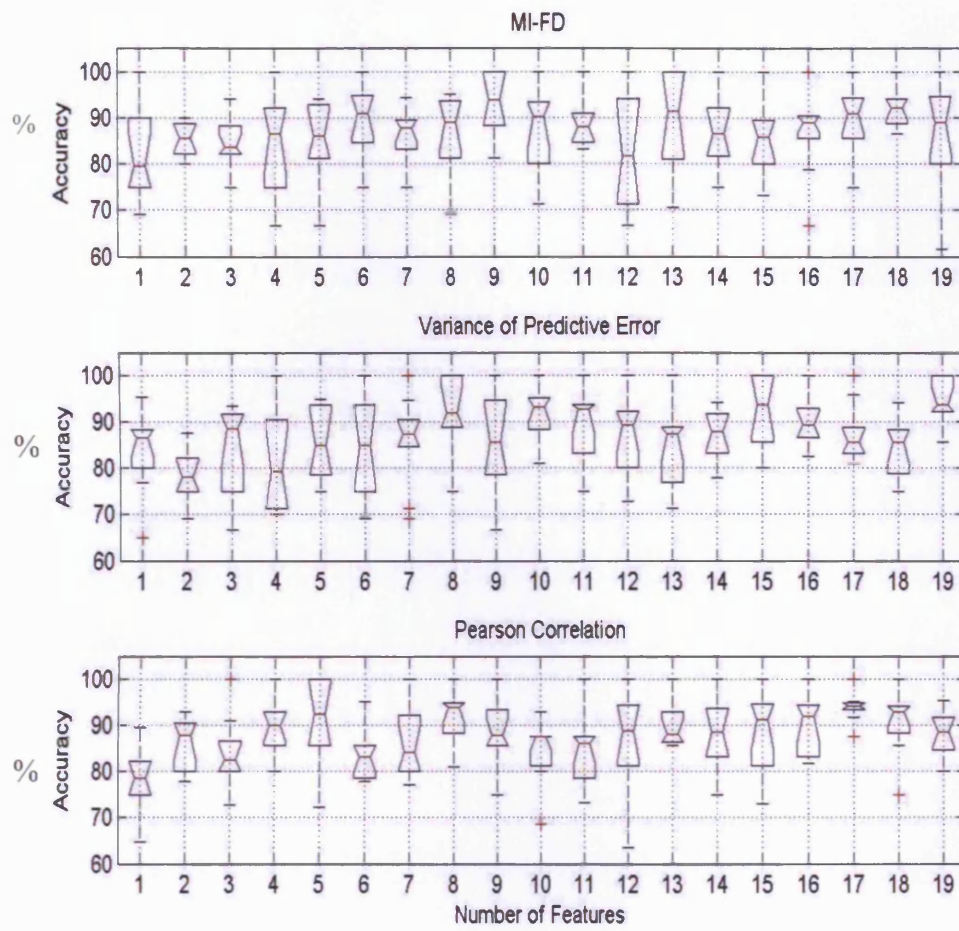


Fig. 5.7: Hepatitis SFS comparison.

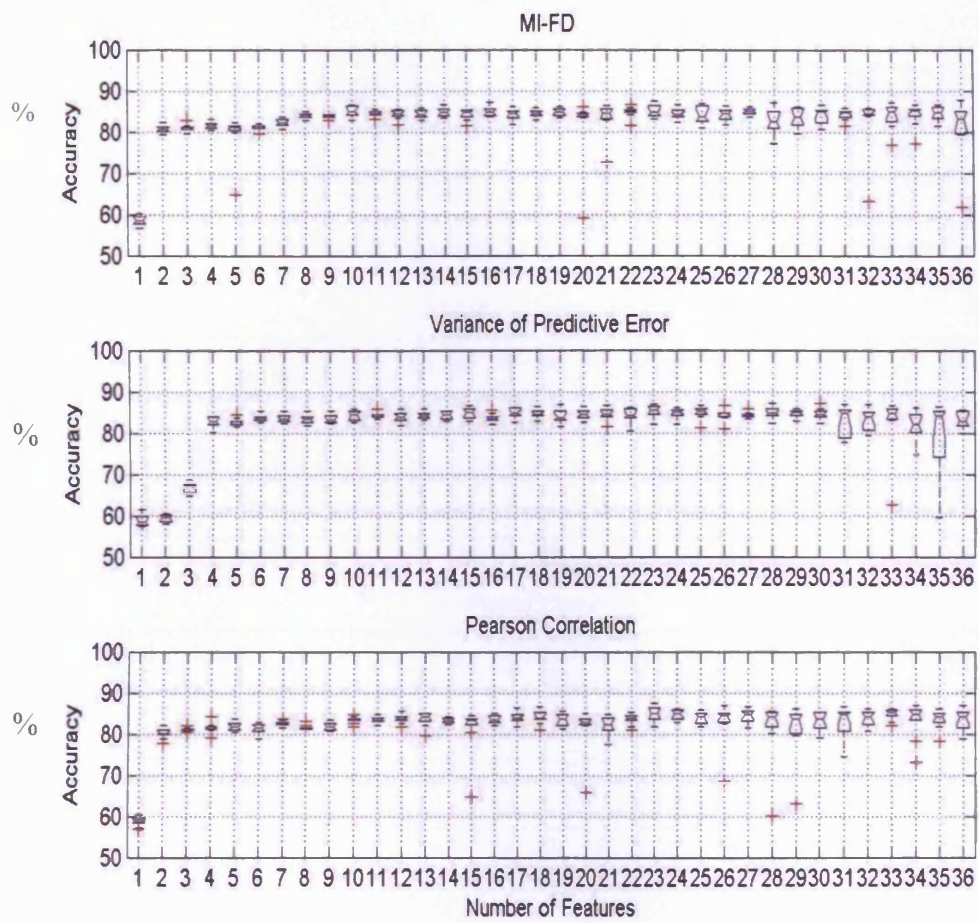


Fig. 5.8: Image Satellite (Statlog version) SFS comparison.

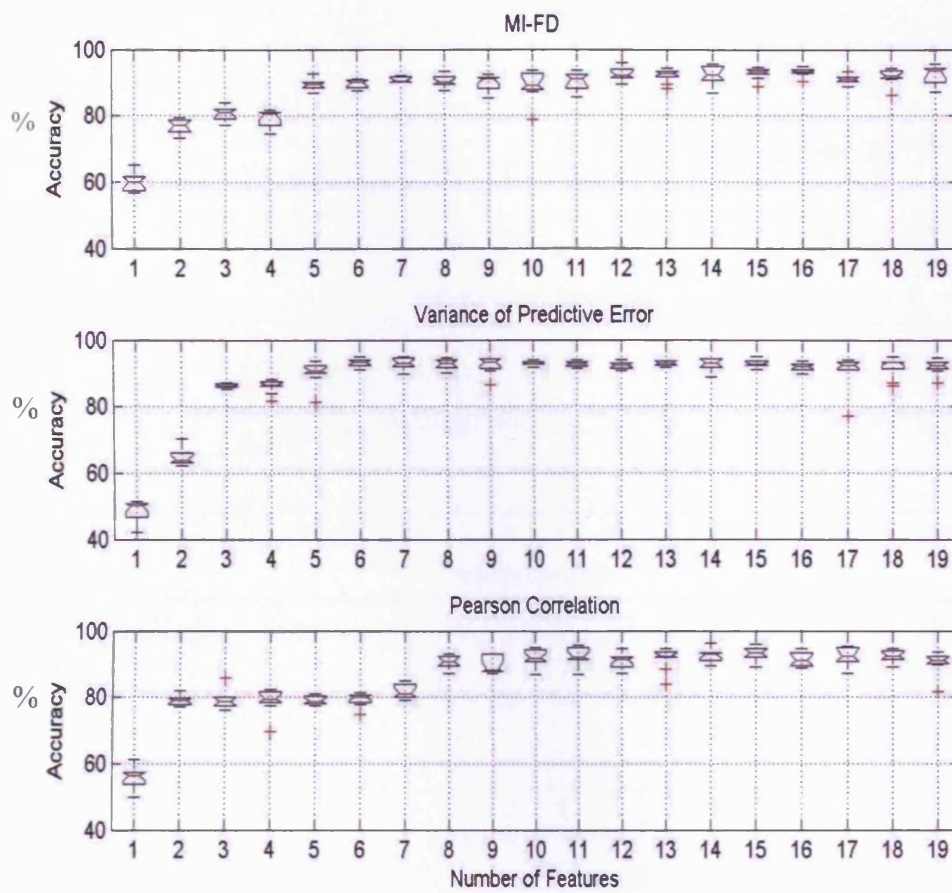


Fig. 5.9: Image Segmentation SFS comparison.

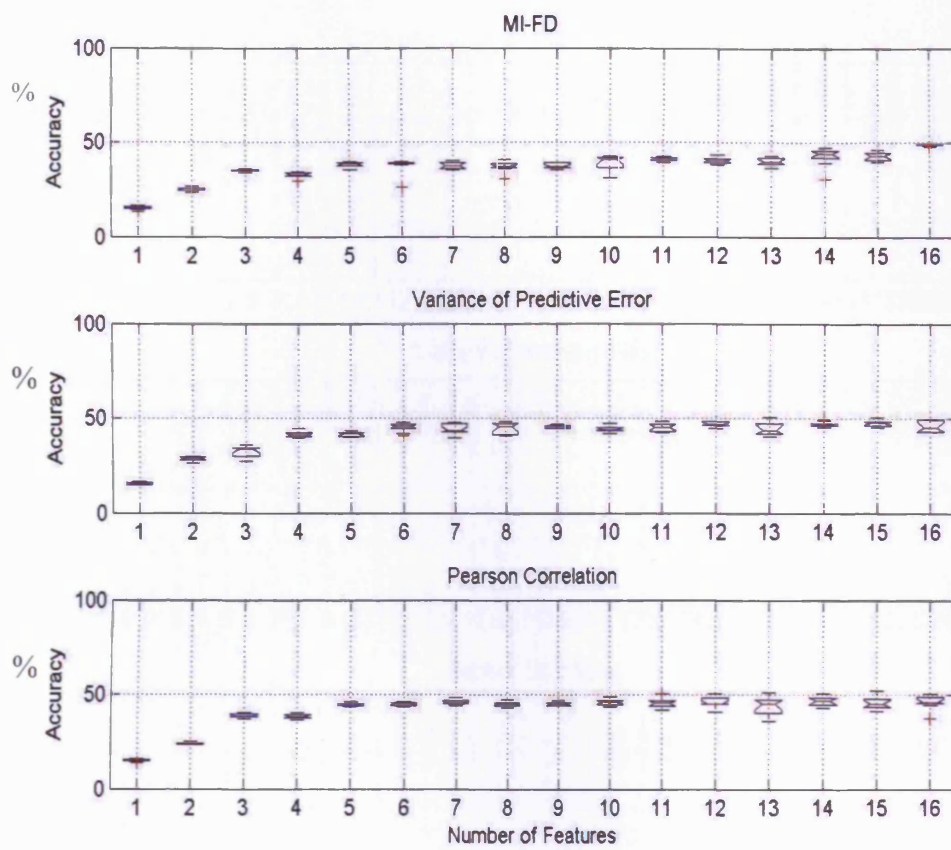


Fig. 5.10: Letter Recognition SFS comparison.

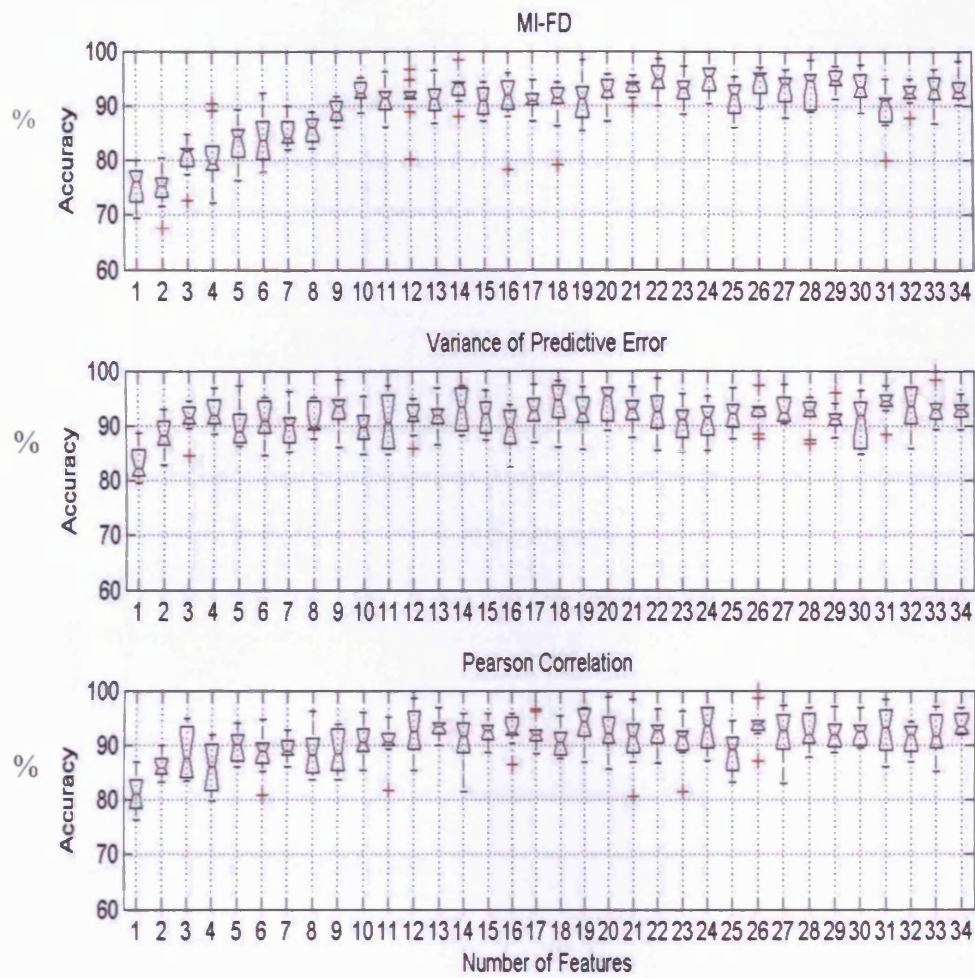


Fig. 5.11: Ionosphere SFS comparison.

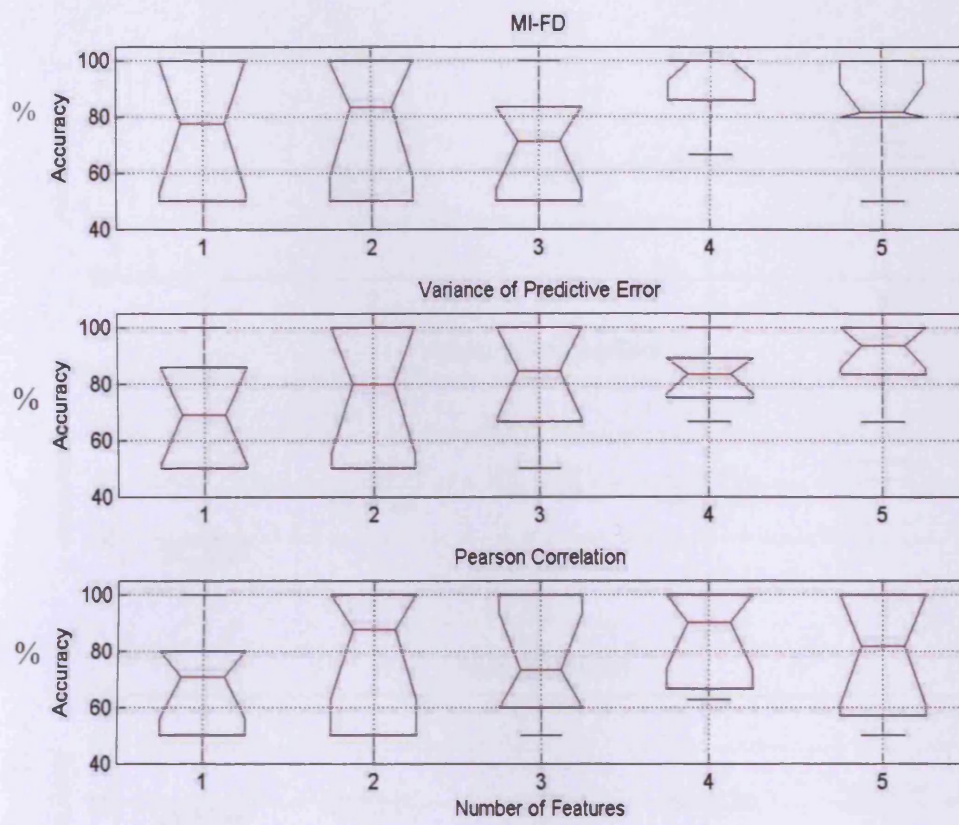


Fig. 5.12: Lenses SFS comparison.



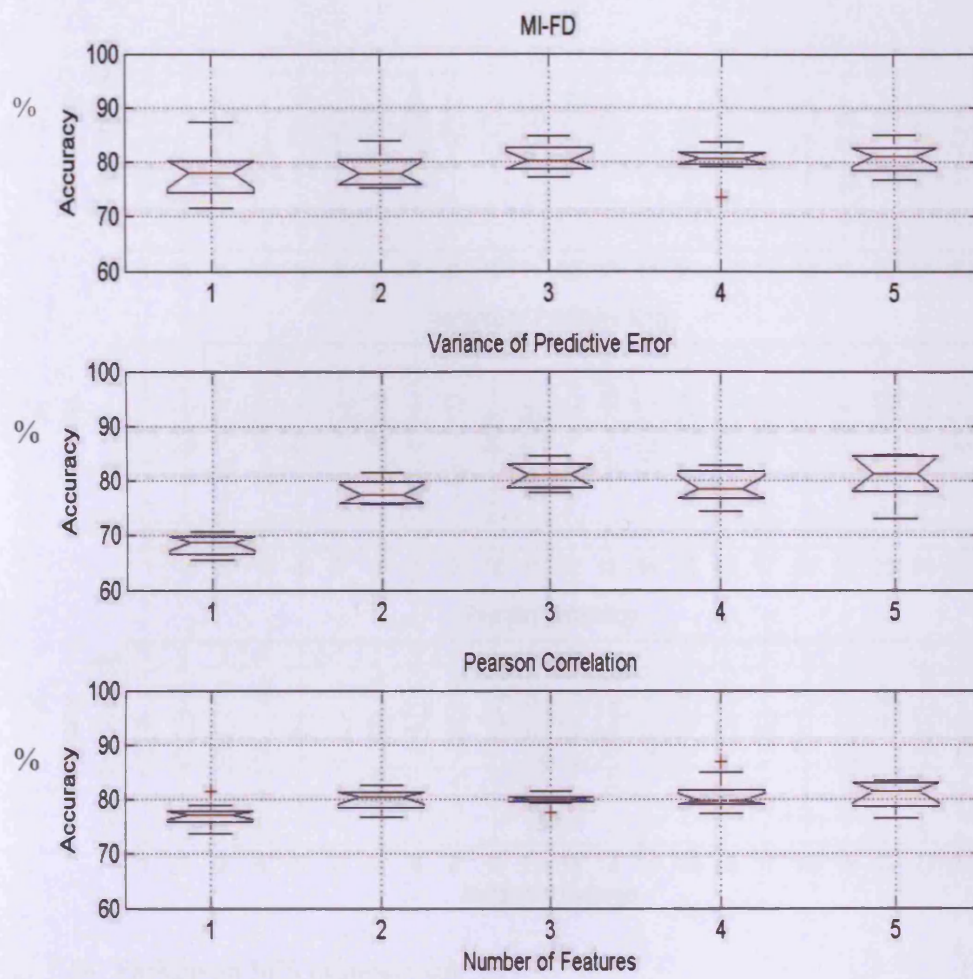


Fig. 5.13: Mammography Masses SFS comparison.

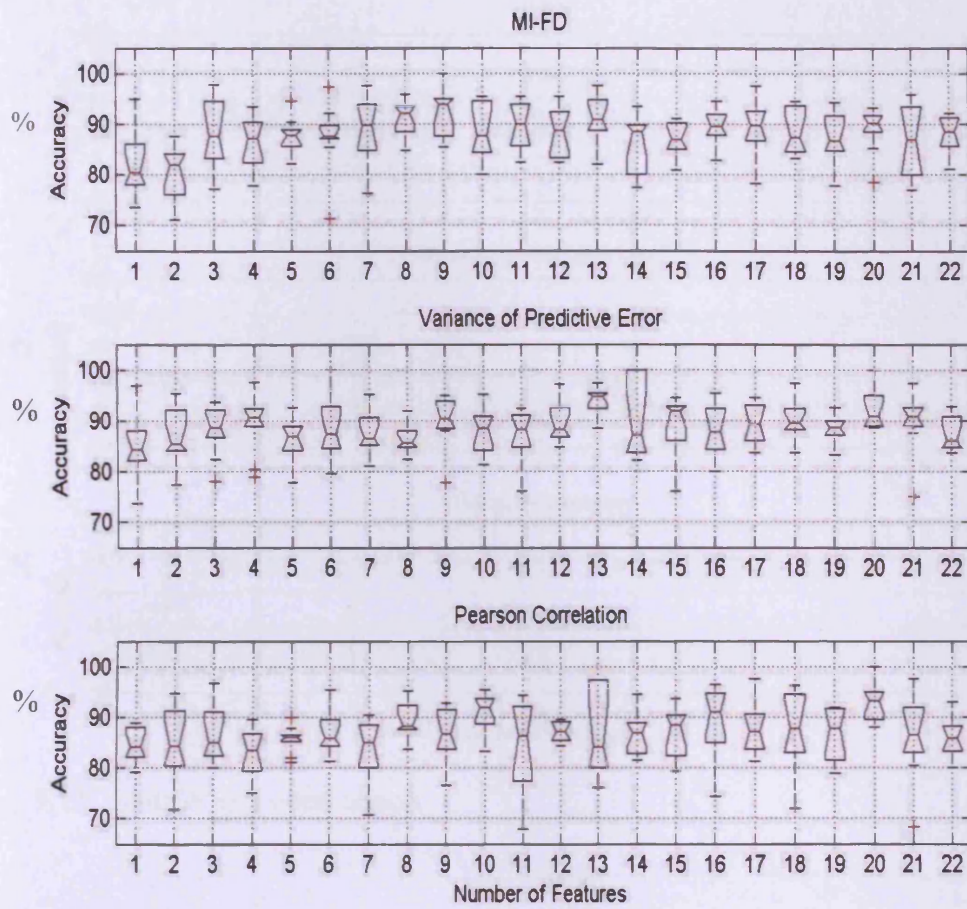


Fig. 5.14: Parkinson SFS comparison.

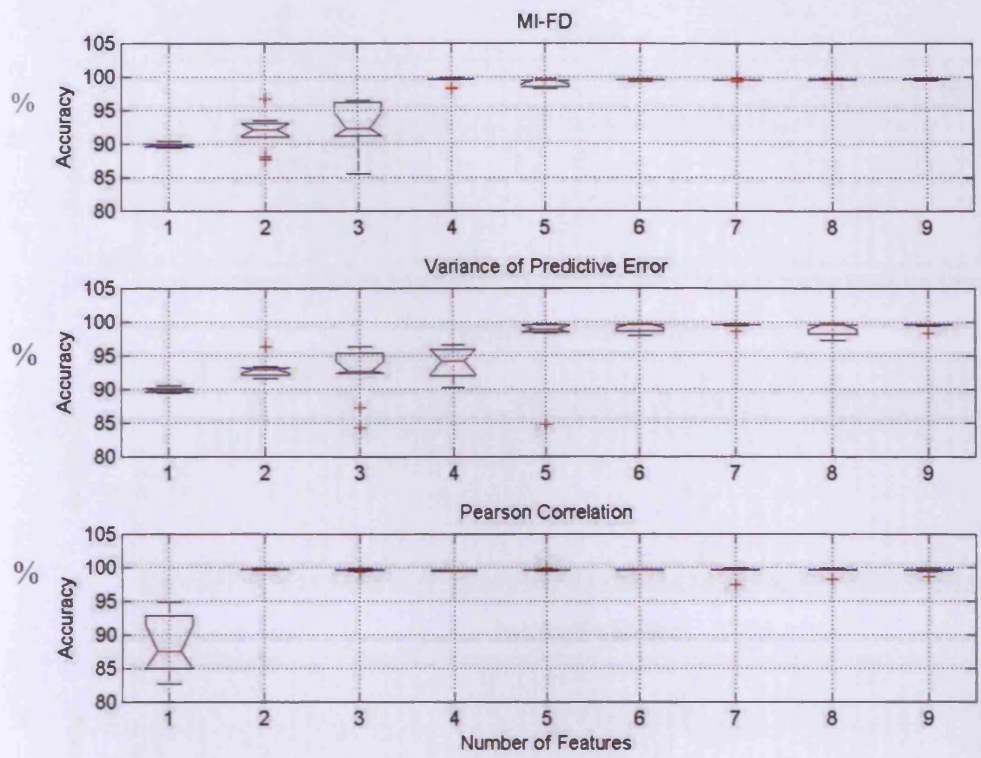


Fig. 5.13: Shuttle SFS comparison.

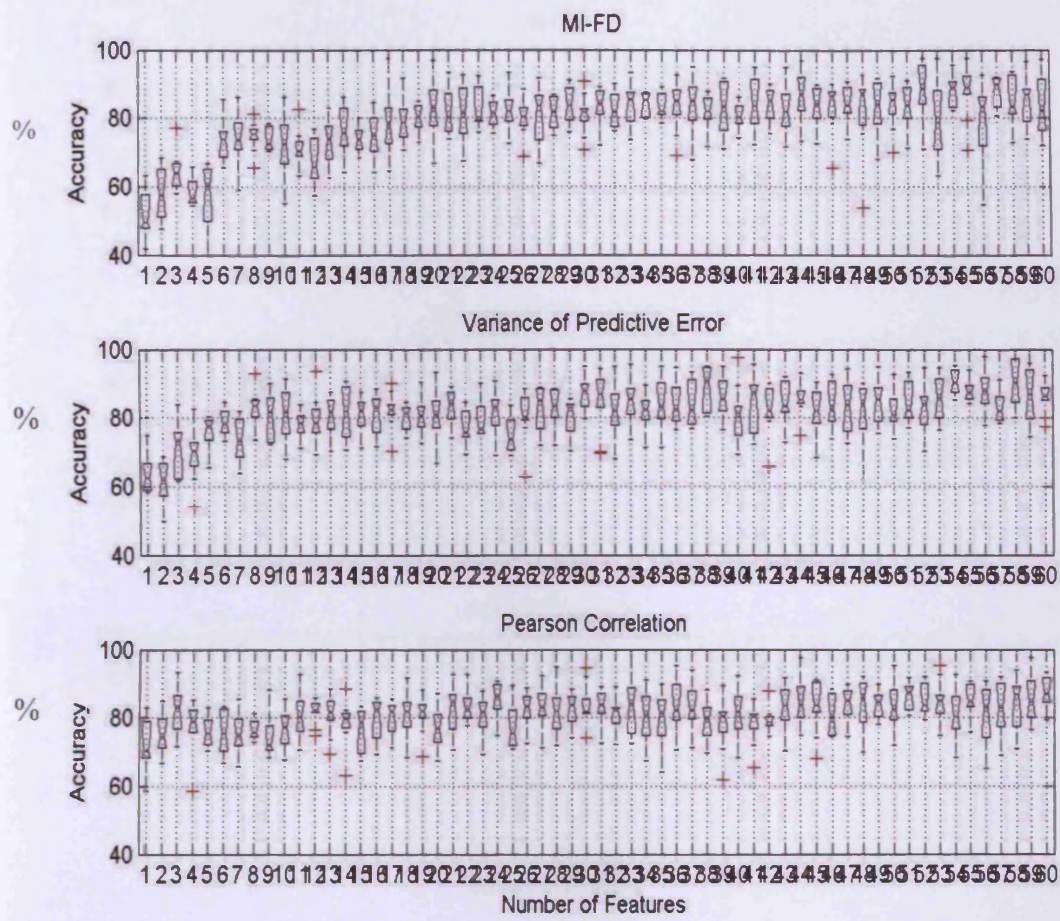


Fig. 5.14: Sonar SFS comparison.

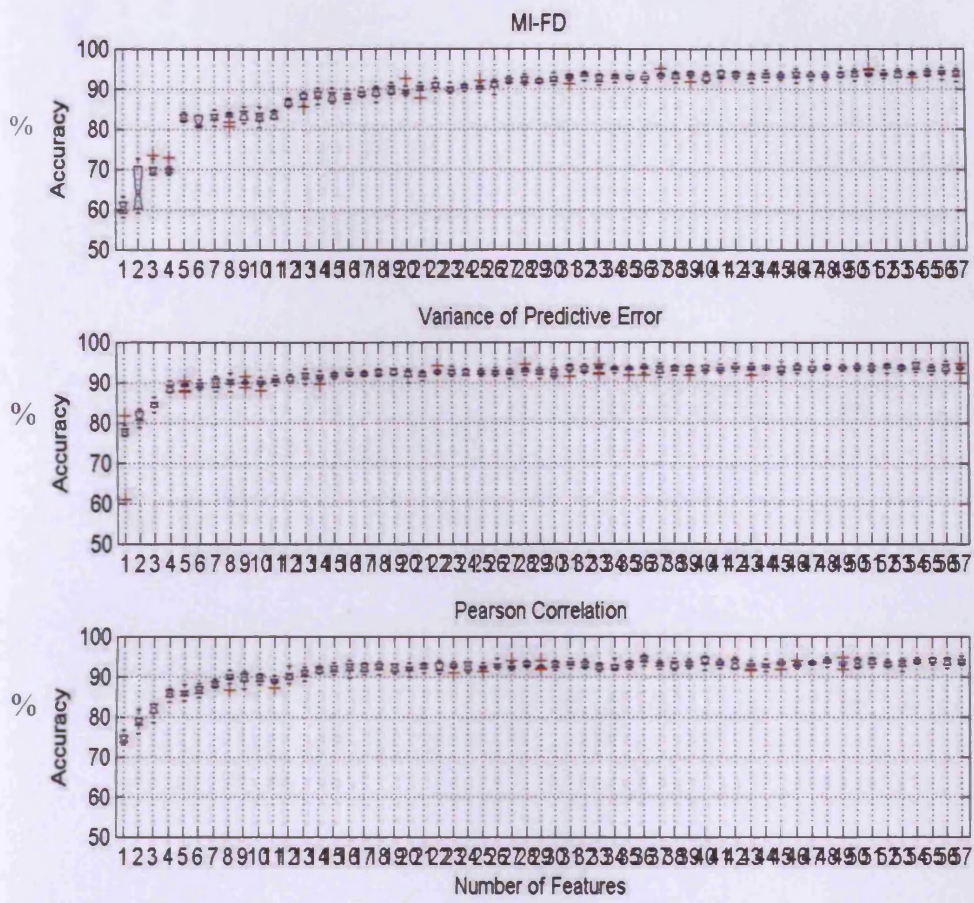


Fig. 5.15: Spambase SFS comparison.

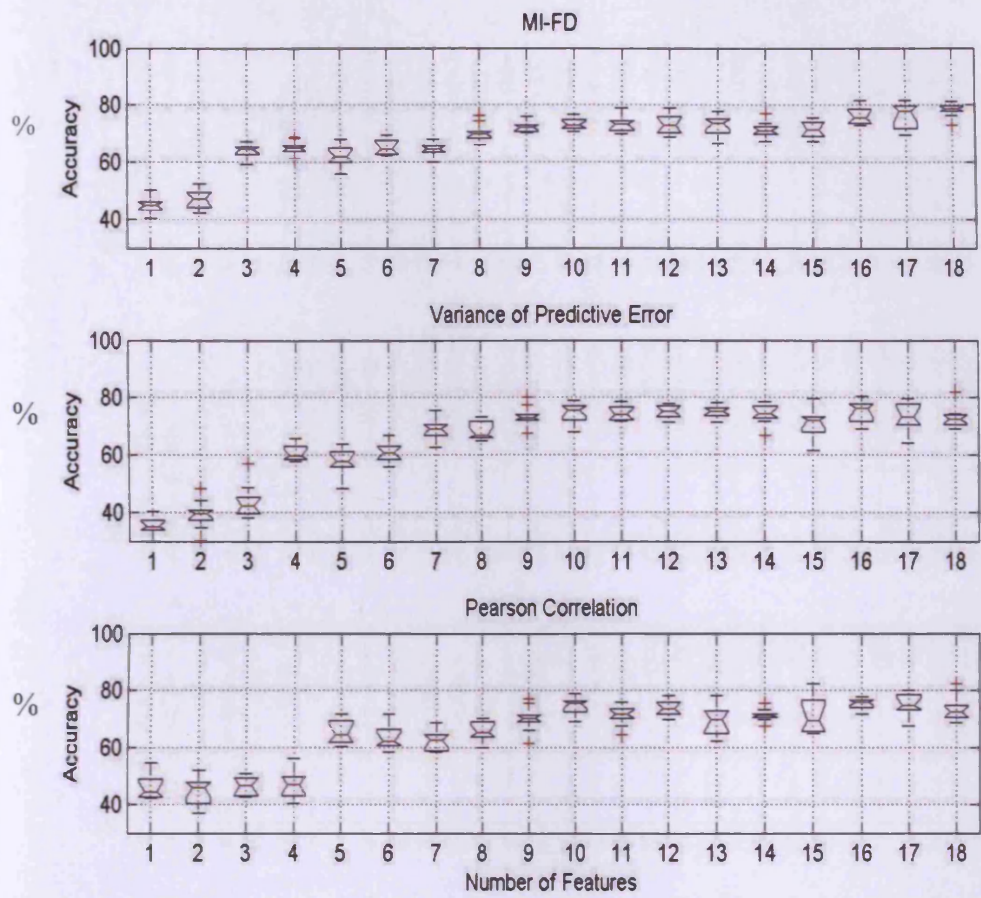


Fig. 5.16: Vehicle SFS comparison.

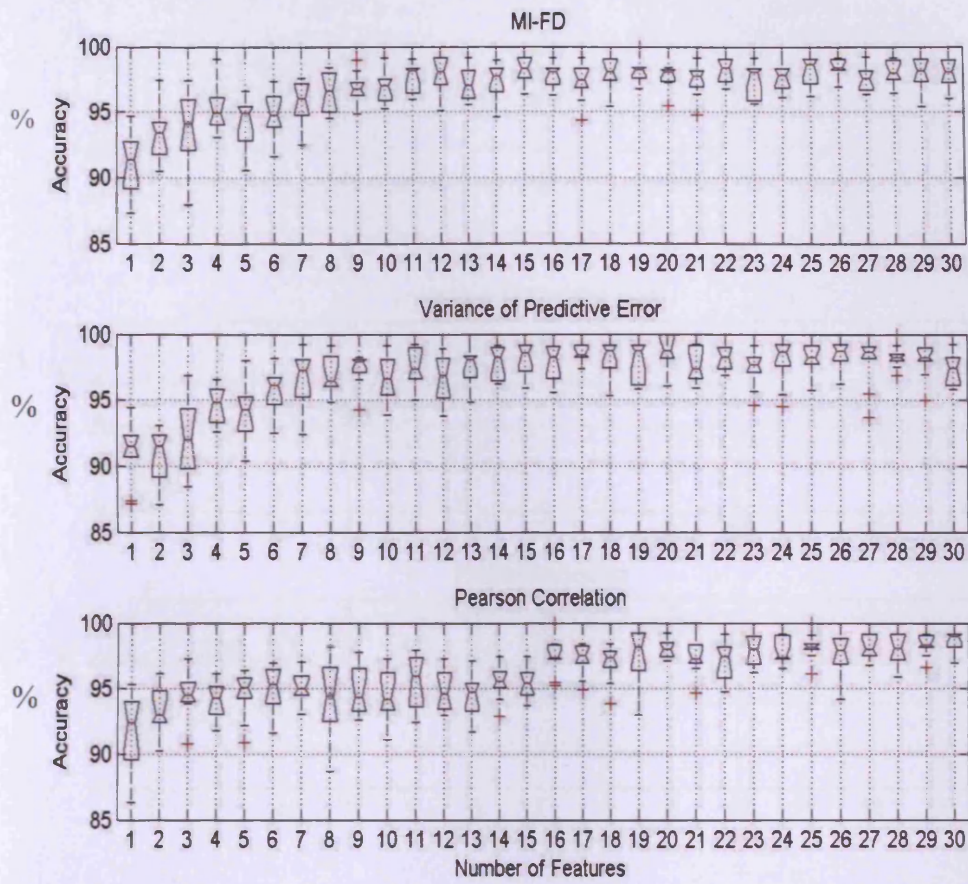


Fig. 5.17: Breast Cancer (wdbc) SFS comparison.

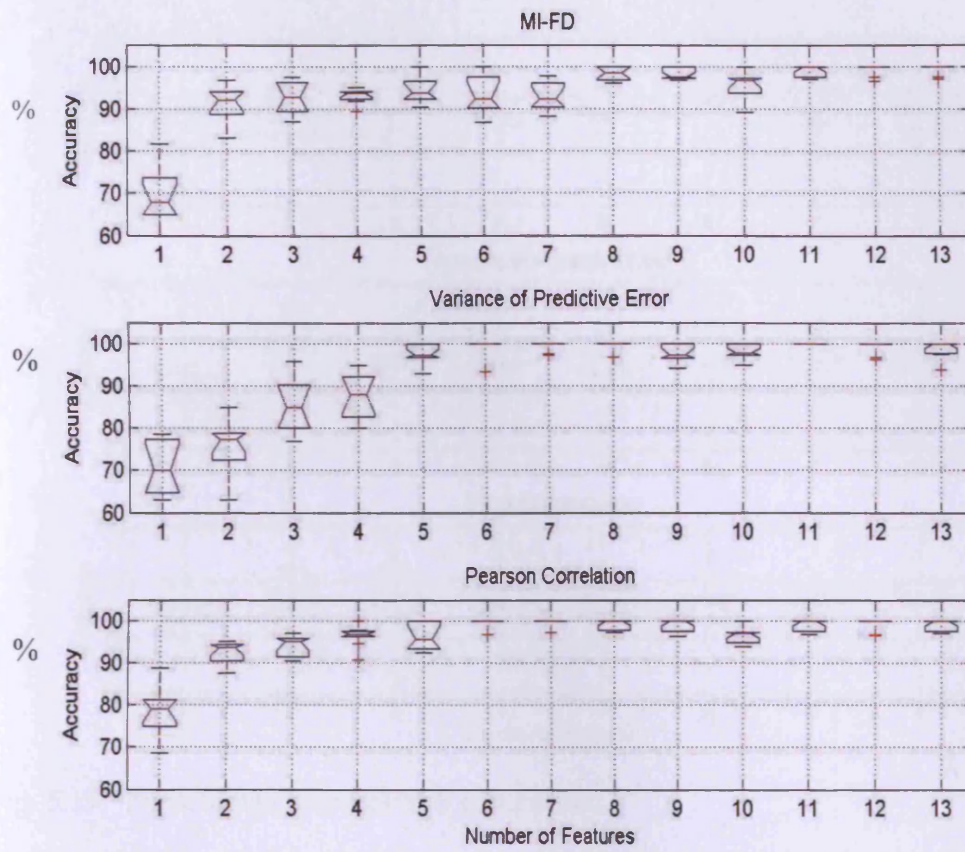


Fig. 5.18: Wine SFS comparison.



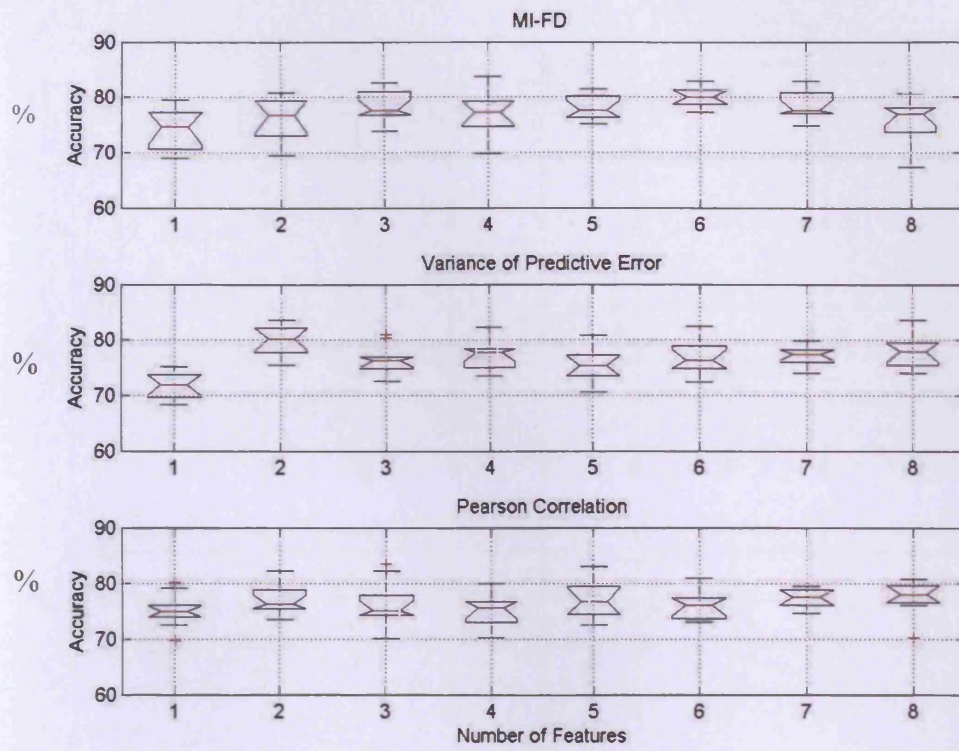


Fig. 5.19: Pima Indian diabetes SFS comparison.

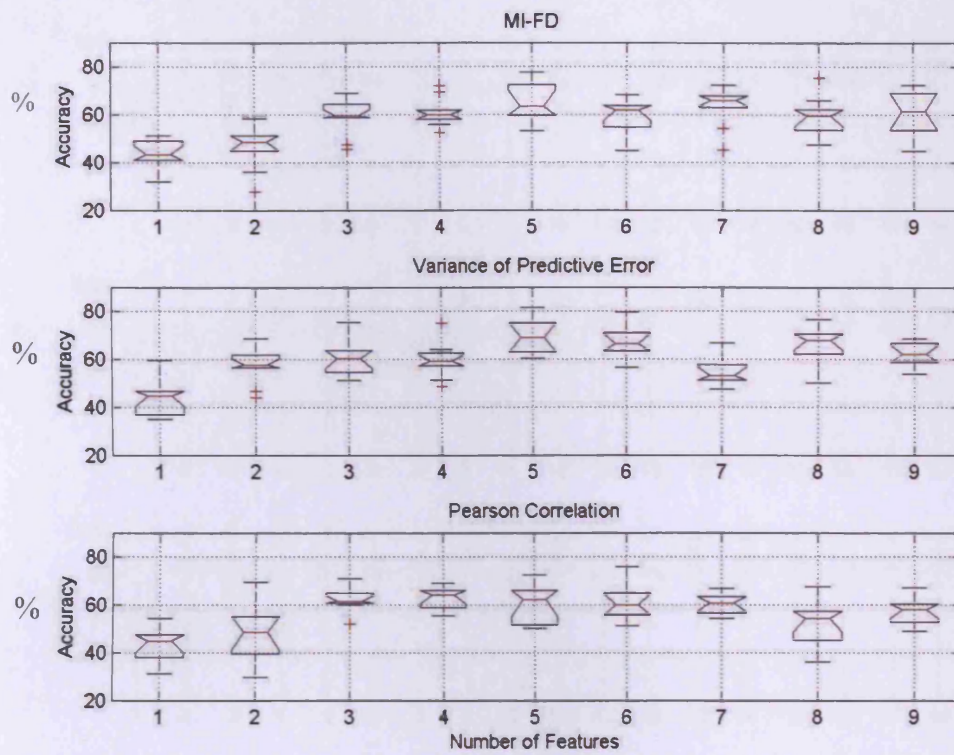


Fig. 5.20: Glass SFS comparison.

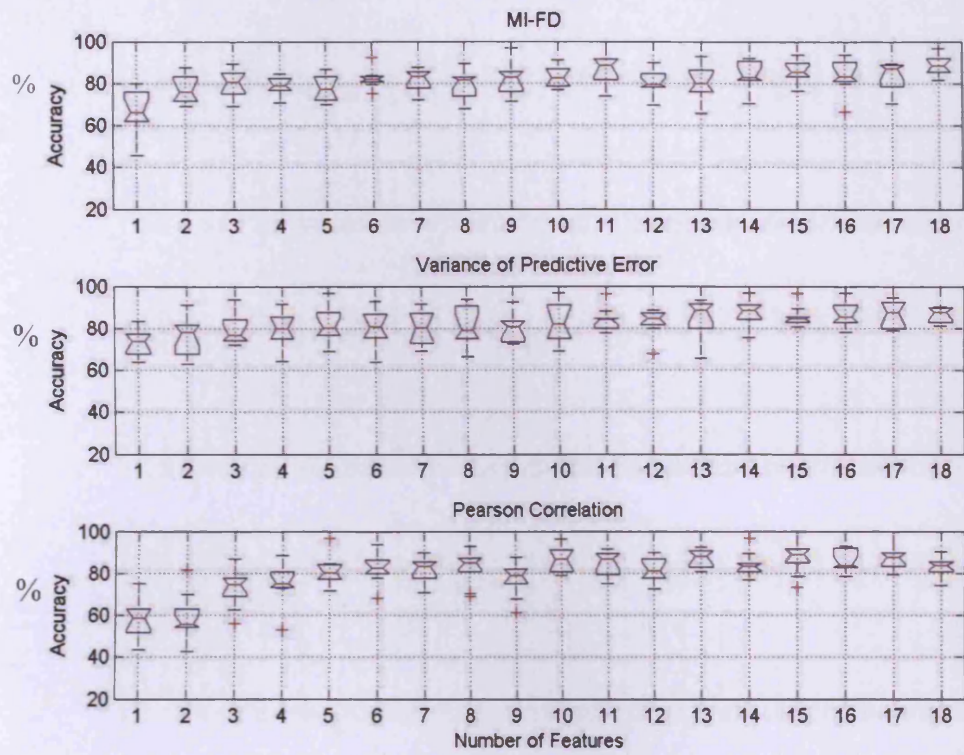


Fig. 5.21: Lymphography SFS comparison.

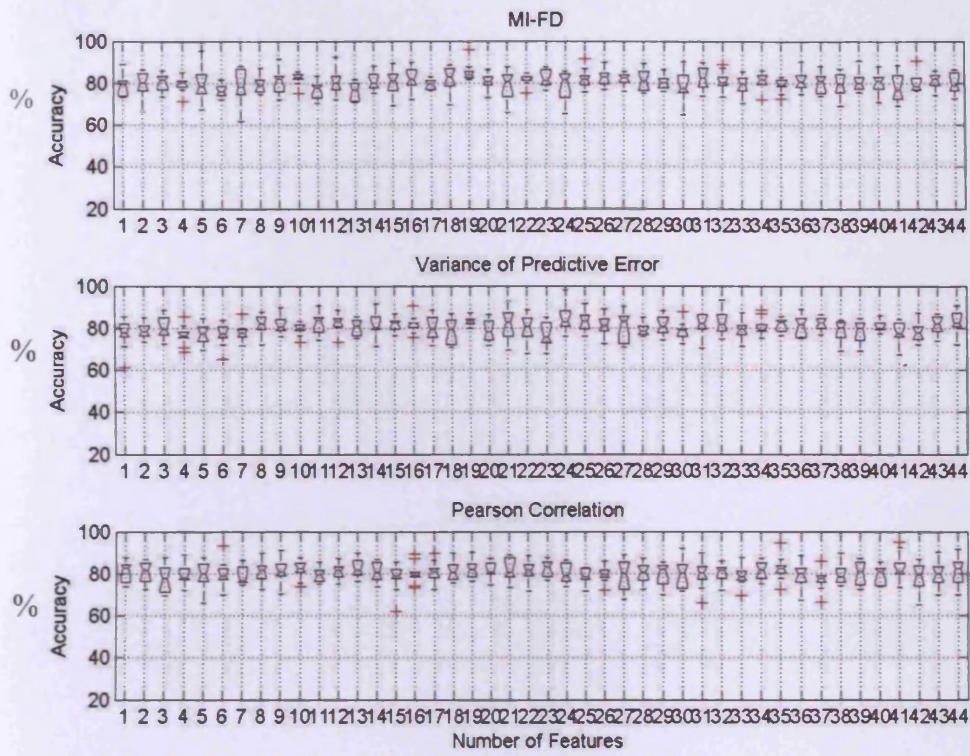


Fig. 5.22: SPECTF Heart SFS comparison.

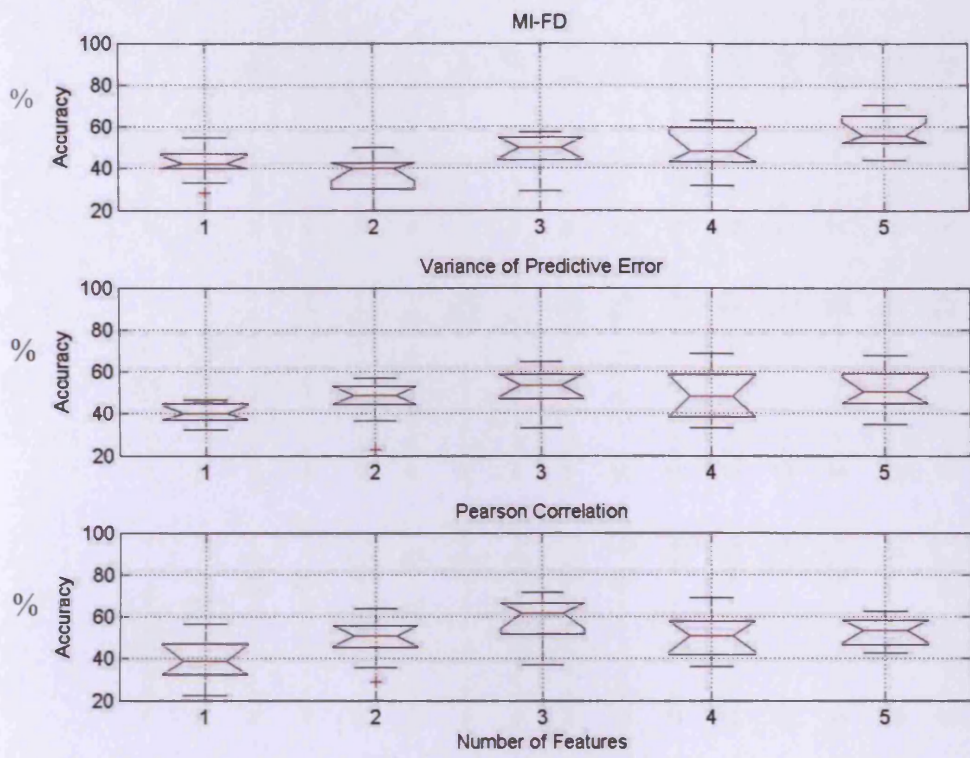


Fig. 5.23: Tae SFS comparison.

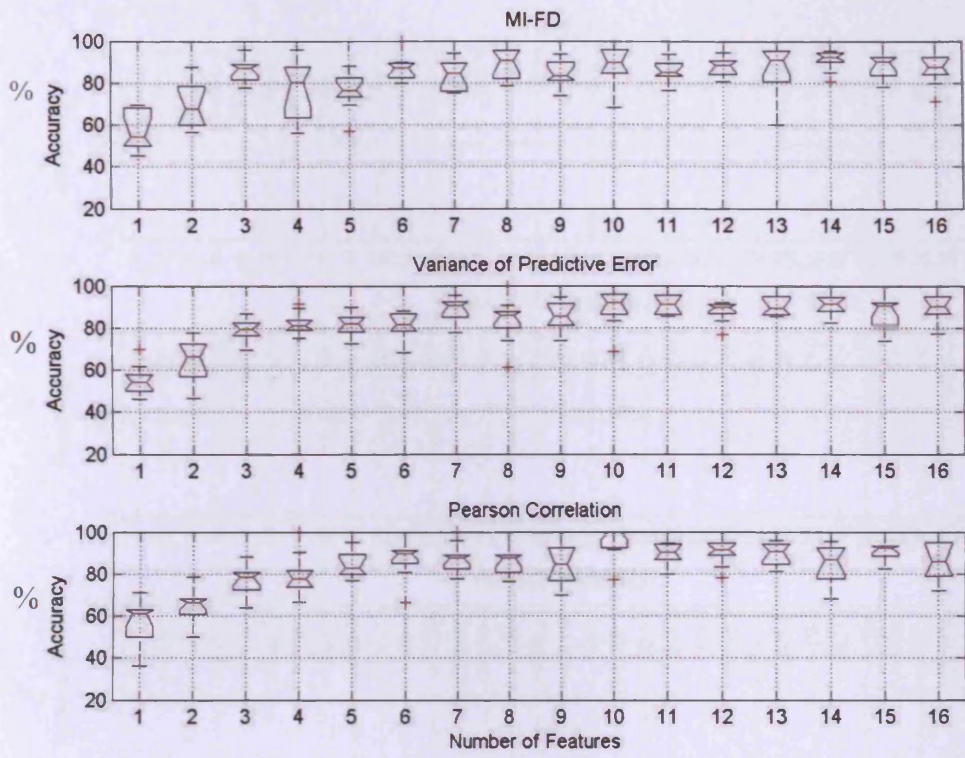


Fig. 5.24: Zoo SFS comparison.

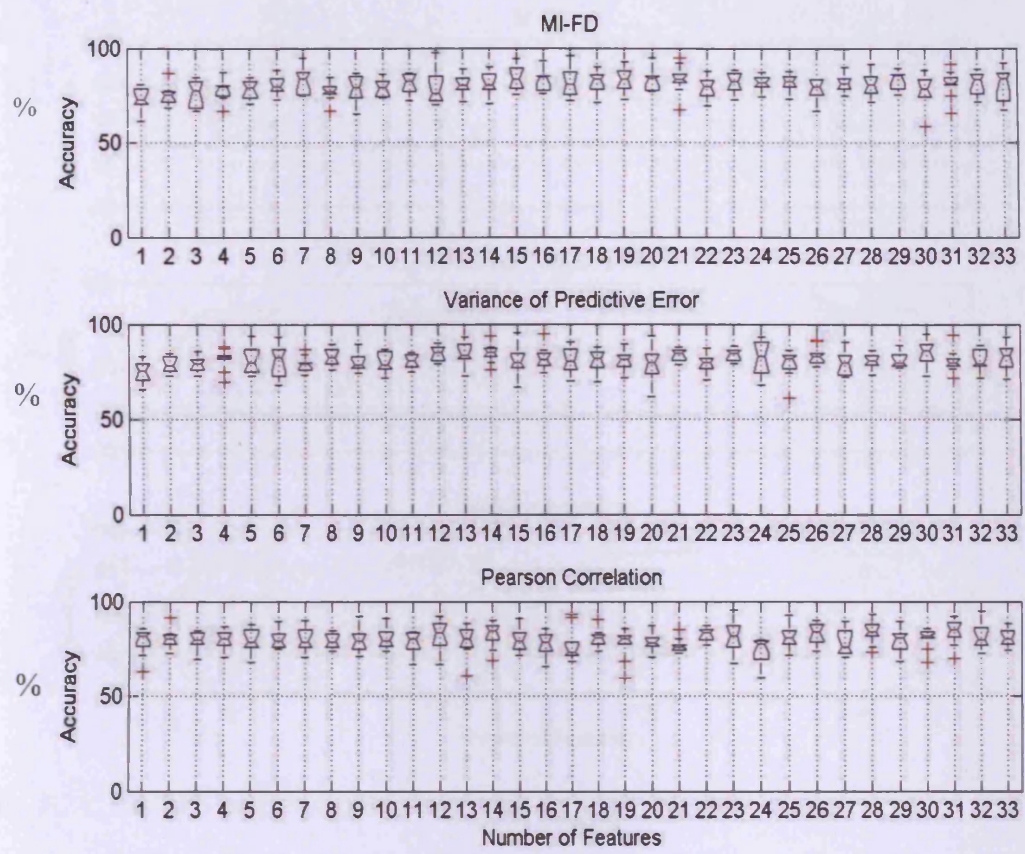


Fig. 5.25: Breast Cancer Wisconsin Prognostic (WPBC) SFS comparison.

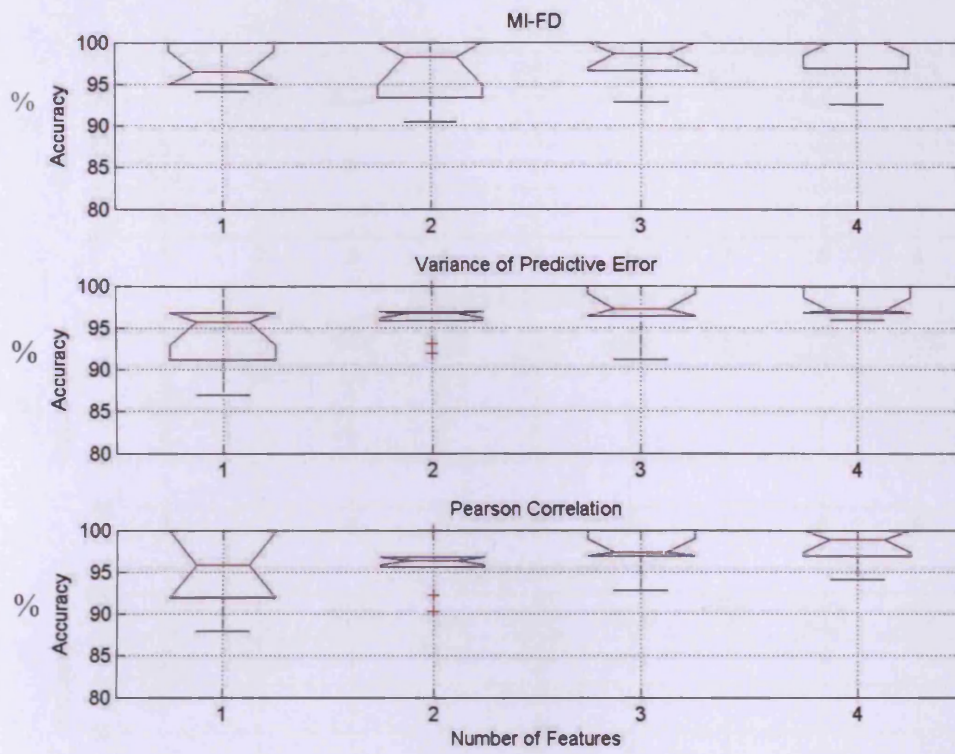


Fig. 5.26: Iris SFS comparison.



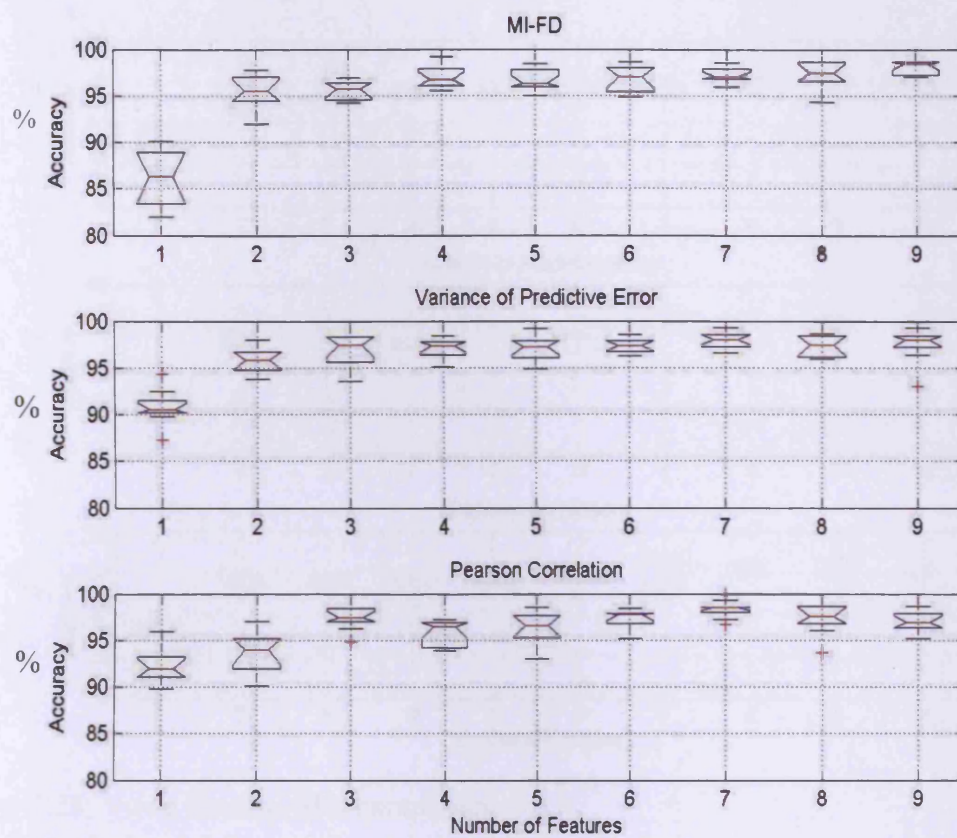


Fig. 5.27: Breast Cancer SFS comparison.

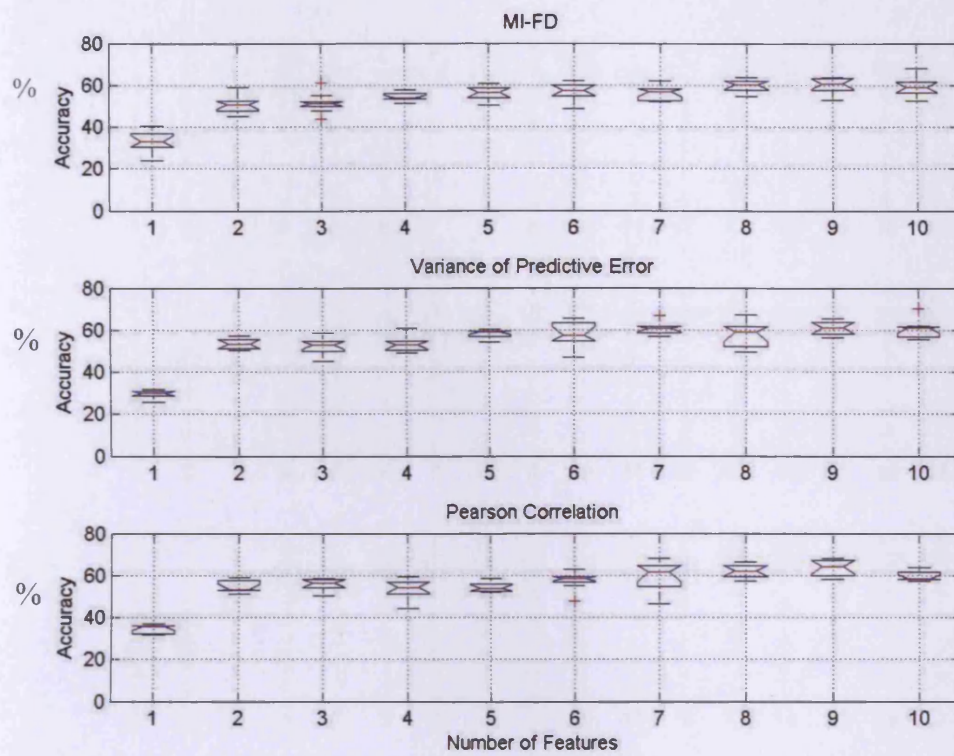


Fig. 5.28: Vowe Context SFS comparison.

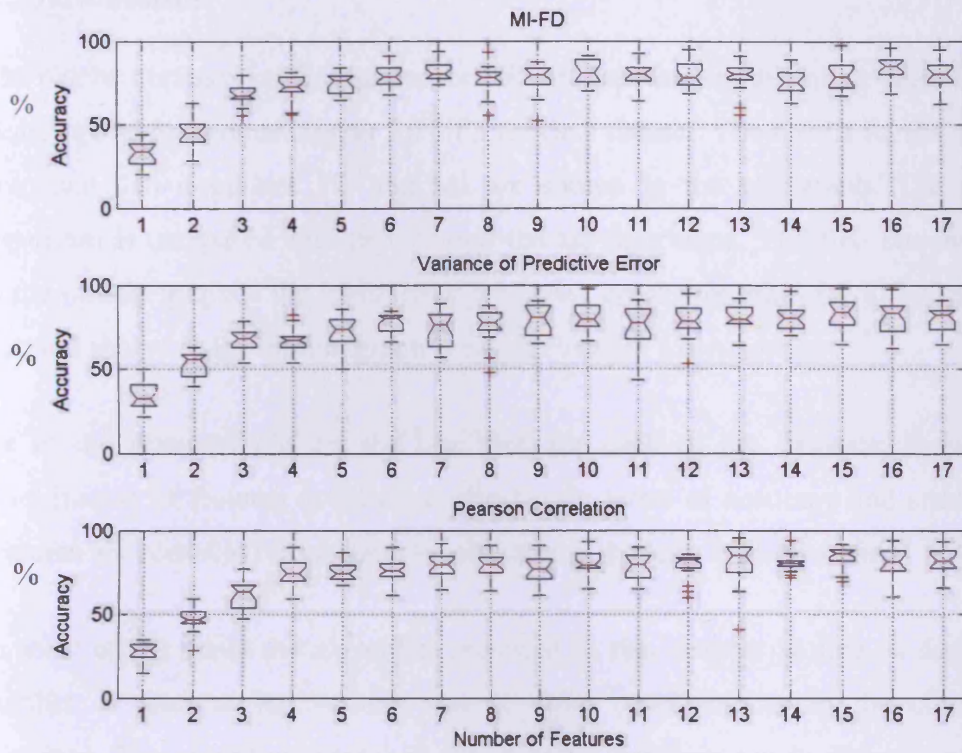


Fig. 5.29: Wood (real application dataset) SFS comparison.

## 5.5 Discussion

The results obtained in this chapter for SFS are represented in similar way, using box plots, as in the previous chapter for UFS for each dataset. The results for the algorithm proposed that combines FD and MI are shown in the top graph. The proposed algorithm is compared with two state-of-the-art algorithms. The first one and shown in the middle graph is the *variance of predictive error* algorithm for SFS. The second one and shown in the bottom graph is called *Pearson Correlation*.

As in the previous chapter the box plot, for each of the datasets, shows which combination of features is the most effective in terms of accuracy and stability. The features are combined from the most effective to the least effective one.

In most of the cases the algorithm proposed in this chapter is able to decrease the number of features maintaining and in some cases improving the classification accuracy. The algorithm proposed shows better performance in datasets such as in *Image Satellite*, *Lenses*, *Mammography masses*, *vehicle*, *wine*, *iris*. The proposed algorithm shows comparable performance to the other SFS techniques as in the cases of *Australian Credit Approval* and *Shuttle* datasets.

## 5.6 Summary

A new algorithm to select useful features for classification applications has been developed. The usefulness of each feature is efficiently measured proposing a simpler and non-heuristic feature selection framework of mRMR. Comparable accuracy is obtained for all of the datasets showing the effectiveness of the subsets selected. The box plots calculated for 10 independent trials on the features selected show that the proposed algorithm has comparable reliability to other state-of-the-art algorithms. The number of features is reduced while keeping effectively the integrity of the information in most of the datasets. The similar nature of MI and FD, both based on mathematical principles, makes the proposed algorithm more suitable for a straightforward analysis of the data. The effect of varying levels of noise in the data on the performance of the proposed fractal based algorithm will be considered in future work.

# Chapter 6

## Conclusion

In this chapter the contributions and conclusions are listed. Directions for further research and improvements for the proposed work are provided.

### 6.1 Contributions

The main contributions of this research are:

- A simple partitioning-hierarchical clustering algorithm that uses the FD as a similarity measure among instances and does not need a bootstrapping by any other different clustering method.
- A divisive analysis technique with a linear computational complexity as it does not need to compute combinatorial dissimilarities among instances to divide data.
- A non-heuristic unsupervised algorithm to select important features, which uses a relatively simple function to measure entropy on datasets based on a fractal (non-Euclidean) similarity measure.
- A supervised algorithm to select important features, which uses a non-heuristic ratio to measure redundancy among features.
- A framework for feature analysis to maximize relevance and minimize redundancy in a parallel way. The framework proposed does not need predictors, so simplifies the feature assessment procedure.

## 6.2 Conclusions

The functionality of the FD, alone and combined, when carrying out data mining tasks such as clustering, supervised and unsupervised feature selection has been shown to be effective. New algorithms to simplify the tasks have been proposed and compared with current state-of-the-art algorithms that have similar characteristics. The following are conclusions for each of the tasks:

- Usually an initialization step in fractal clustering is extremely important in determining a good quality of clusters. In the method proposed, for fractal clustering, the initialization step has been eliminated, giving an adequate recognition of natural clusters in artificial and benchmark datasets.
- The proposed fractal clustering algorithm uses similarities instead of traditional Euclidean distances to divide the data into sub-clusters. The divisive analysis maintains a sub-cluster balance, by splitting in *half range* the sub-clusters, which is a factor for a good performance when the sub-clusters are re-merged to create final clusters.
- Important features can be found by detecting changes of entropy in data. Usually these changes in entropy are detected using sequential search techniques. These techniques are not needed in the algorithm for UFS developed in this research because it is simpler than the current state of the art algorithms. The simplicity of the proposed algorithm relies on a new function to calculate entropy in the data which uses FD to measure similarity among instances.

Because the FD has the property to quantify the complexity of the whole data, the proposed algorithm needs fewer loops to calculate the entropy, and no search technique to determine the important features. The algorithm shows comparable and in some cases better results than the current state-of-the-art algorithm, the latter having to use Euclidean distance to calculate similarity among instances and a sequential backward selection to find the important features.

The method is useful to remove irrelevant features and helps a clustering method to find groups in data, with a reduced set of features. Comparing supervised and unsupervised selected features on a clustering method gives a better insight of the performance of the unsupervised selection methodology, as in the SFS the important features are selected using *true* group labels for each of the instances.

- The FD has been incorporated efficiently with the MI to produce a more straightforward FS algorithm to find relevant and non-redundant features. The proposed algorithm produces smaller feature sets which deliver stable and sometimes superior classification accuracies to the compared algorithms.
- In this research the FD was used to develop solutions to a fundamental problem in data mining which is: *searching the whole space* in order to guarantee global maximisation of a criterion function. The proposed fractal solutions, for clustering and unsupervised and supervised FS, give non-heuristic solutions to alleviate the computational burden that search techniques produce.

### 6.3 Suggestions for Future Research

- Further work should be directed at the complexity of the FD algorithm, in terms of features, in order to create a linear computational complexity algorithm.
- Developing a method to calculate the number of clusters for a given dataset. A clustering algorithm that computes the number of clusters automatically and simultaneously with a FS procedure. FS is a worthwhile line to pursue to find more compact feature subsets while maintaining the quality of the clusters.
- Investigating local UFS methods that select different subsets of features for different clusters appear particularly useful, since they can be applied when,

known in advance. All the features in the dataset are necessary for the clustering task.

- Extending the method of SFS to deal with regression problems in which the class contains continuous values.
- Investigating the effect of varying levels of noise in the data on the performance of the fractal base algorithm.



# Appendix A

## Back Propagation Algorithm

The training of a multilayer perceptron uses a method called back propagation of error, based on the generalized delta rule (Rumelhart). For each record presented to the network during training, information (in the form of input fields) feeds forward through the network to generate a prediction from the output layer. This prediction is compared to the recorded output value for the training record, and the difference between the predicted and actual output(s) is propagated backward through the network to adjust the connection weights to improve the prediction for similar patterns.

In the feed-forward calculation the information flows through the network as follows:

Input neurons have their activations set to the values of the encoded input fields. The activation of each neuron in a hidden or output layer is calculated as:

$$a_i = \sigma(\sum_j w_{ij} o_j)$$

where  $a_i$  is the activation of neuron  $i$ ,  $j$  is the set of neurons in the preceding layer,  $w_{ij}$  is the weight of the connection between neuron  $i$  and neuron  $j$ ,  $o_j$  is the output of neuron  $j$ , and  $\sigma(x)$  is the sigmoid or logistic transfer function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

## Back-propagation calculation

When the training starts all the weights in the network are set to random values in the interval  $-0.5 \leq w_{ij} \leq 0.5$ .

Records are presented in cycles (also called epochs), where each cycle involves presenting  $n$  randomly selected training records to the network, where  $n$  is the number of records in the training data. Because of the random selection process, in any particular cycle some training records may be presented more than once and others may not be presented at all. For each record, information flows through the network to generate a prediction, as described above. The prediction is compared to the target value found in the training data for the current record, and that difference is propagated back through the network to update the weights. To be more precise, the change value  $\Delta w$  for updating the weights is calculated as:

$$\Delta w_{ij}(n+1) = \eta \delta_{pj} o_{pi} + \alpha \Delta w_{ij}(n),$$

where  $\eta$  is the learning rate parameter,  $\delta_{pj}$  is the propagated error, described below,  $o_{pi}$  is the output of neuron  $i$  for record  $p$ ,  $\alpha$  is the momentum parameters, and  $\Delta w_{ji}(n)$  is the change value for  $w_{ji}$  at the previous cycle.

The value of  $\alpha$  is fixed during training, but the value of  $\eta$  varies across cycles of training.  $\eta$  starts at the user-specified initial eta, decreases logarithmically to the value of low eta, reverts to the high eta value, and then decreases again to low eta. The value of  $\eta$  is calculated as

$$\eta(t) = \eta(t-1) \exp\left(\frac{\log\left(\frac{\eta_{low}}{\eta_{high}}\right)}{d}\right)$$

where  $d$  is the user-specified number of eta decay cycles. If  $\eta(t-1) < \eta_{low}$  then  $\eta(t)$  is set to  $\eta_{high}$ .  $\eta$  and continues to cycle thusly until training is complete. The back-propagated error value  $\delta_{pj}$  is calculated based on where the connection lies in the network. For connections to output neurons it is calculated as

$$\delta_{pj} = (t_{pj} - o_{pj}) o_{pj} (1 - o_{pj})$$

where  $k$  is the set of neurons to which neurons  $j$ 's output is connected,  $w_{kj}$  is the weight between the current neuron and the neuron  $k$ , and  $\delta_{pk}$  is the propagated error for that weight for the current input record. Weights are updated immediately as each record is presented to the network during training.

# Appendix B

## Supervised Feature Selection Algorithms

### Pearson's correlation chi-square

Pearson's correlation chi-square is a test of independence between  $X$  and  $Y$  that involves the difference between the observed and expected frequencies. The expected cell frequencies under the null hypothesis of independence are estimated by. Under the null hypothesis, Pearson's chi-square converges asymptotically to a chi-square distribution with degrees of freedom  $d = (I-1)(J-1)$ .

The  $p$  value based on Pearson's chi-square  $\chi^2$  is calculated by  $p \text{ value} = \text{Prob}(\chi^2_d > \chi^2)$ , where

$$\chi^2 = \frac{\sum_{i=1}^I \sum_{j=1}^J (N_{ij} - \hat{N}_{ij})^2}{\hat{N}_{ij}}$$

Predictors are ranked by the following rules.

1. Sort the predictors by  $p$  value in the ascending order
2. If ties occur, sort by chi-square in descending order.
3. If ties still occur, sort by degree of freedom  $d$  in ascending order.
4. If ties still occur, sort by the data file order.

The likelihood ratio chi-square is a test of independence between  $X$  and  $Y$  that involves the ratio between the observed and expected frequencies. The expected cell frequencies under the null hypothesis of independence are estimated by. Under the null hypothesis, the likelihood ratio chi-square converges asymptotically to a chi-square distribution with degrees of freedom  $d = (I-1)(J-1)$ .

The  $p$  value based on likelihood ratio chi-square  $G^2$  is calculated by  $p \text{ value} = \text{Prob}(\chi^2_d > G^2)$ , where

$$G^2 = 2 \sum_{i=1}^I \sum_{j=1}^J G^2_{ij} = \begin{cases} \frac{N_{ij} \ln(N_{ij})}{N_{ij}} & N_{ij} > 0; \\ else = 0 & \end{cases}$$

Predictors are ranked according to the same rules as those for the  $p$  value based on Pearson's chi-square.

## Variance of Predictive Error

The variable of importance algorithm requires  $n_{round}$   $\times$   $n_{sample\_per\_round}$  data generated. For each round, the corresponding accuracy for each sub-model (feature subset) is determined. Feature  $X_j$  for each of the  $j$  predictors; across all rounds, average accuracy and variance can be calculated.

- For each round,  $n_{sample}$  random cases are generated as follows:
  - $Y$  is assigned a random value based on the prior probabilities  $\pi_k$ .
  - Each  $X_j$  is randomly assigned based on conditional probabilities

$$\hat{P}(X_j | Y = y)$$

Within a round, each of the  $X_j$  features are excluded from the model, and the accuracy is calculated based on the generated test data for each submodel in turn.

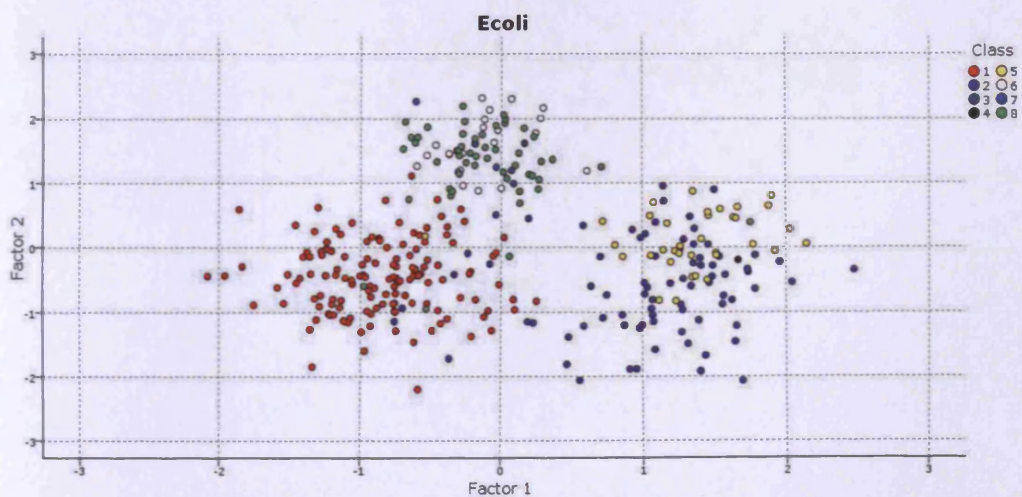
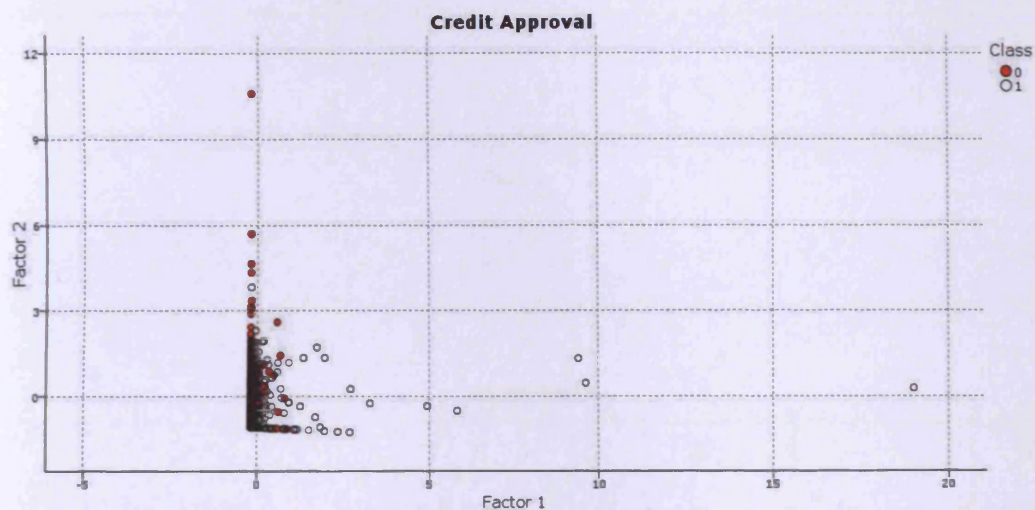
- The accuracies for each round are calculated by comparing the submodel's predicted value for each case's generated outcome  $y$ ,

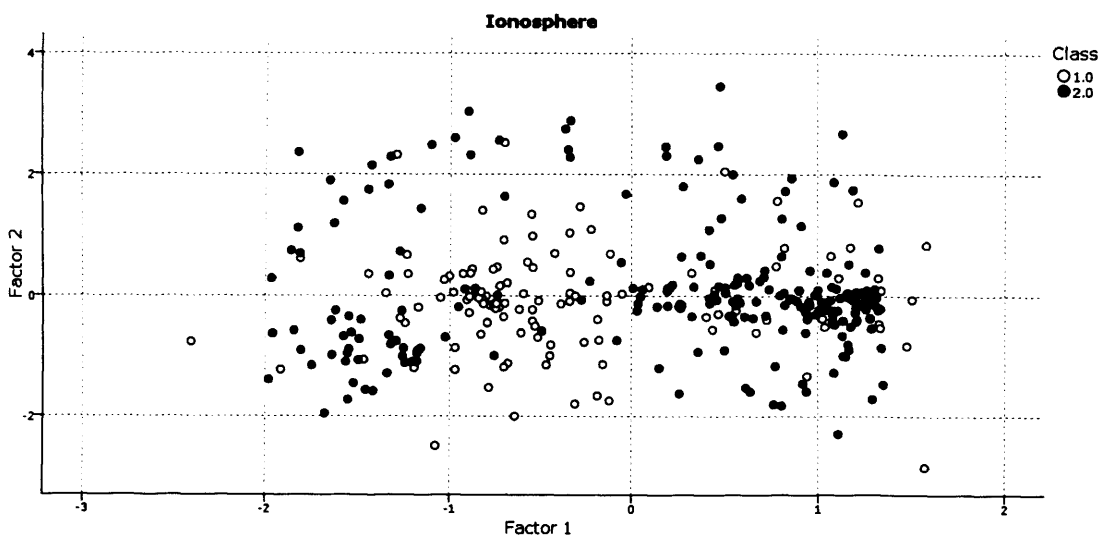
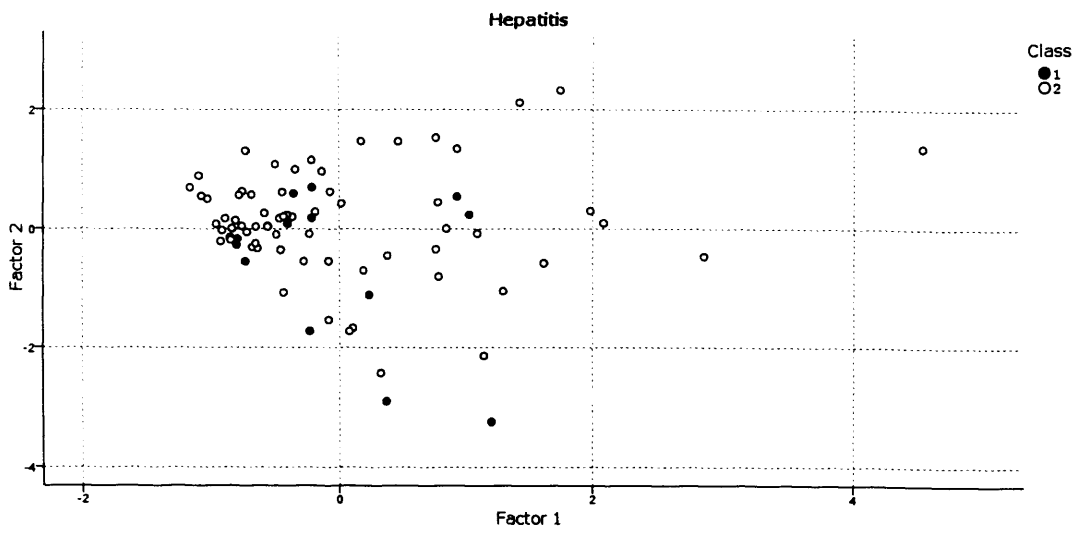
$$P_{accuracy\_without\_xj} = \frac{n_{correct\_without\_xj}}{n_{sample}}, \text{ for each of the } j \text{ submodels.}$$

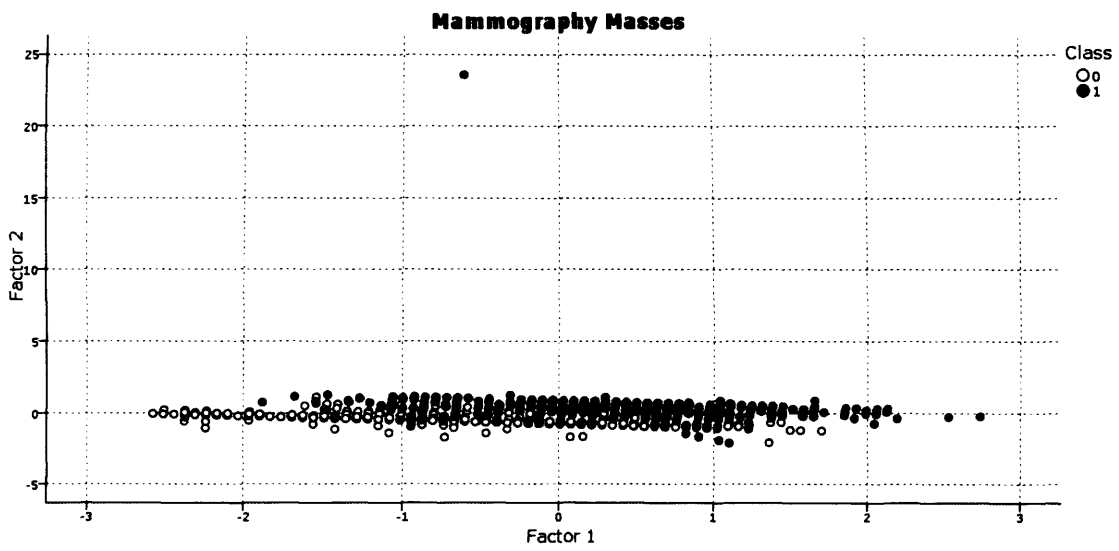
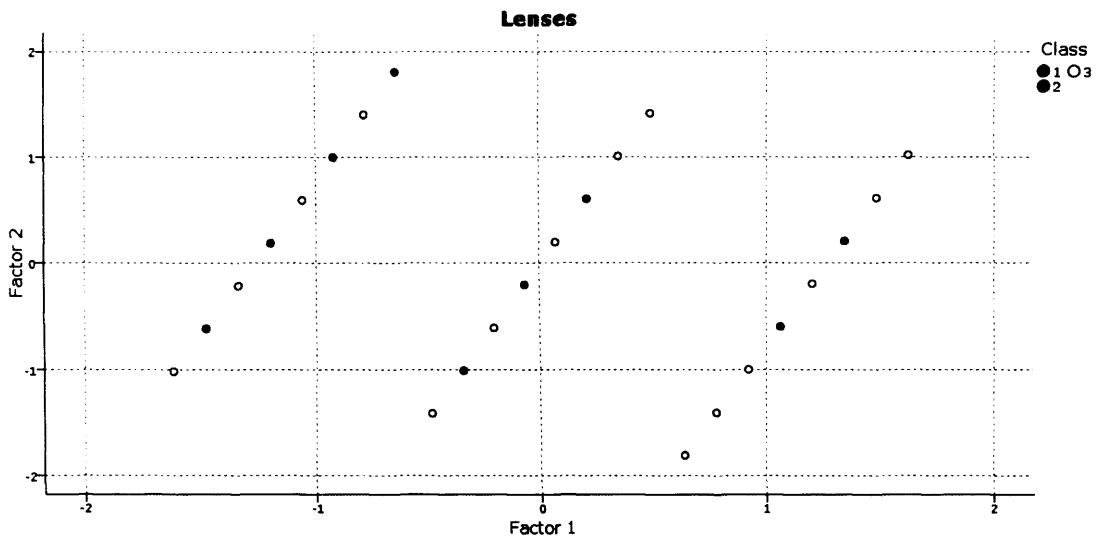
- The mean and variance of the estimated accuracy are calculated across rounds for each sub-model. For each variable, the importance is measured as the difference between the accuracy of the full model and the mean accuracy for the sub-models that excluded the variable.

# Appendix C

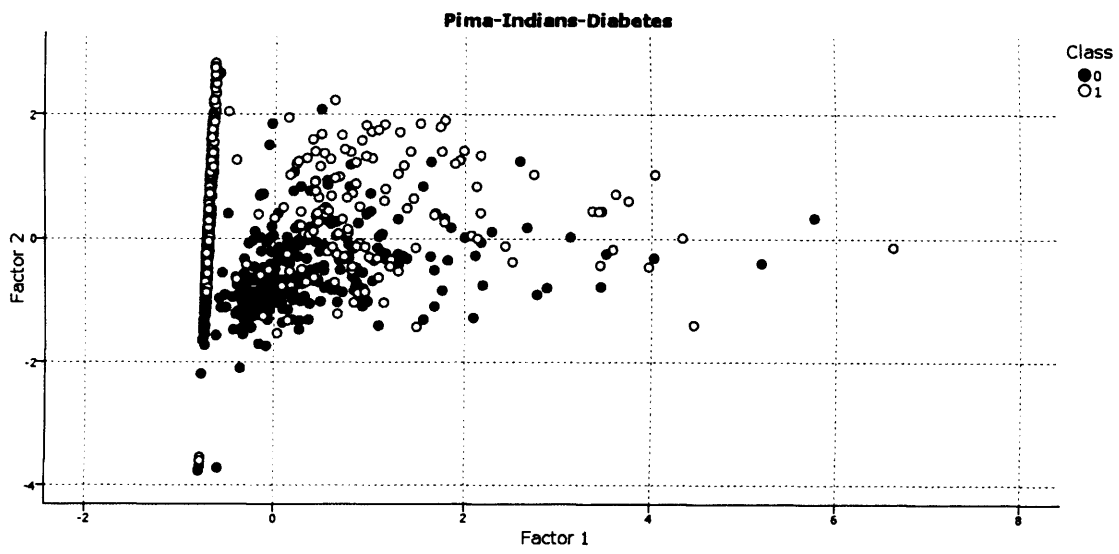
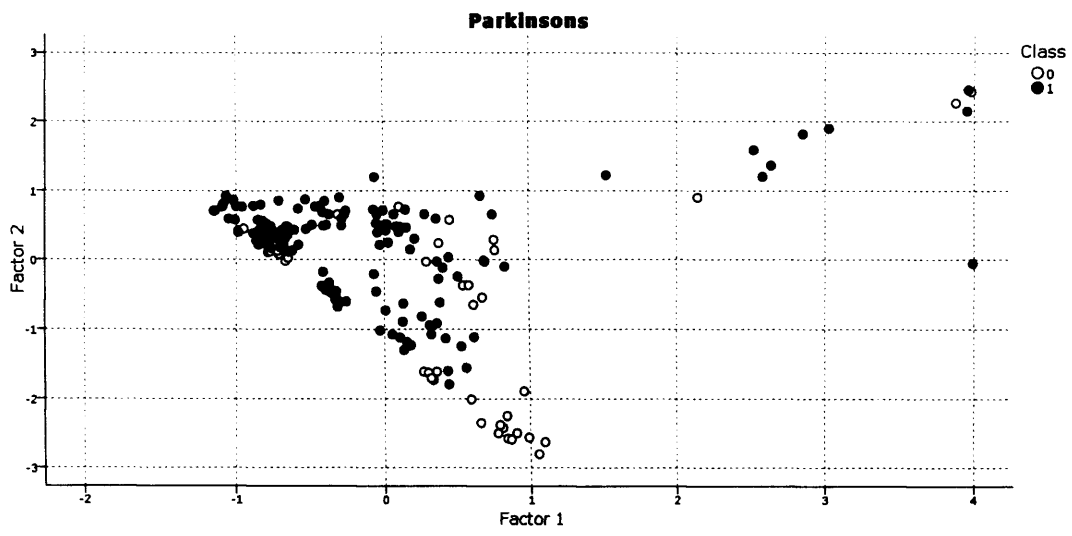
## Two Principal Component Projections of the Benchmark datasets used

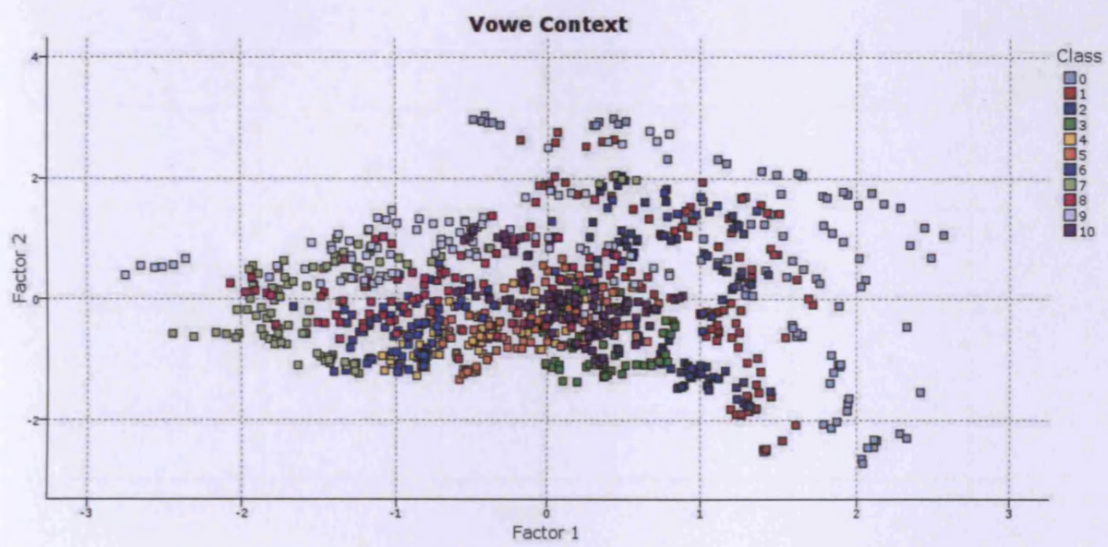
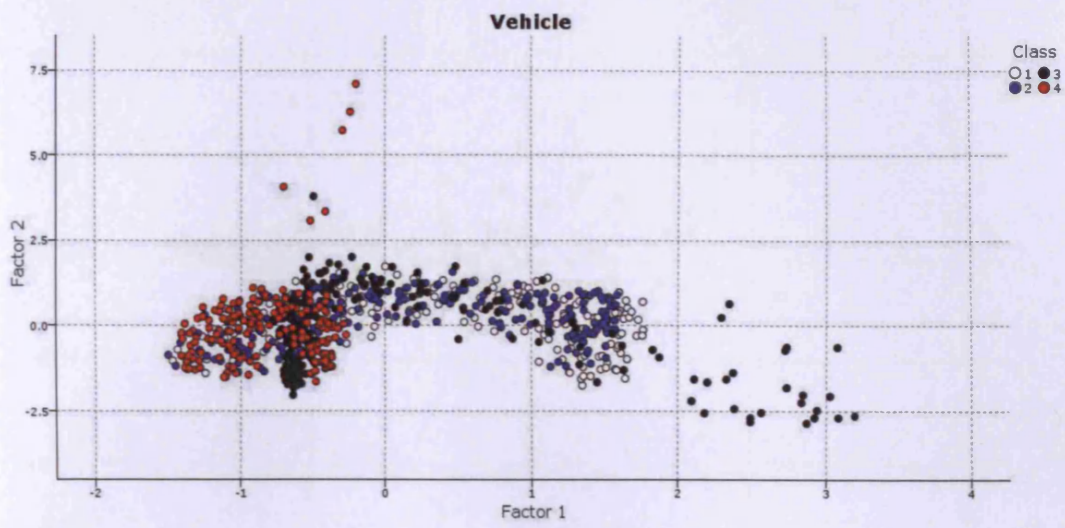


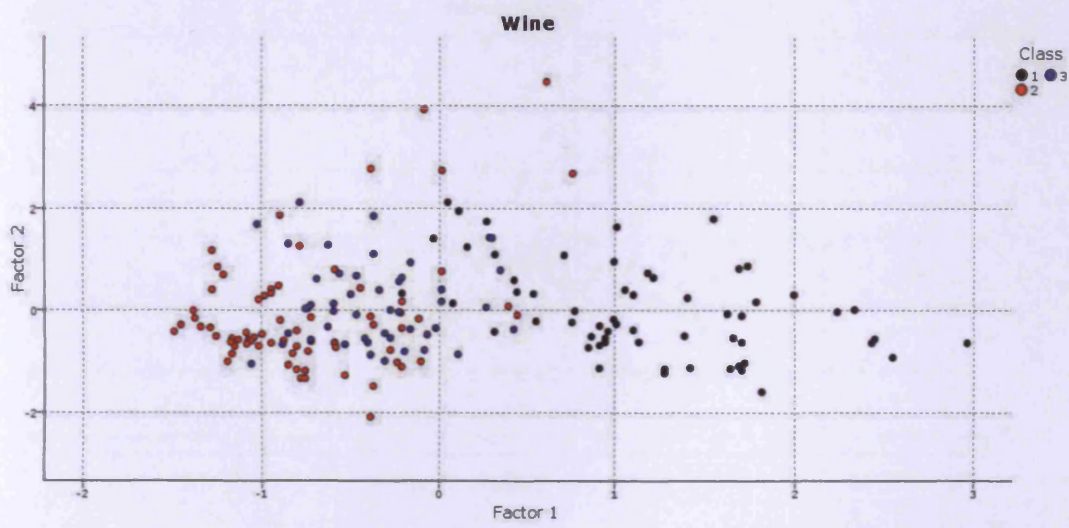
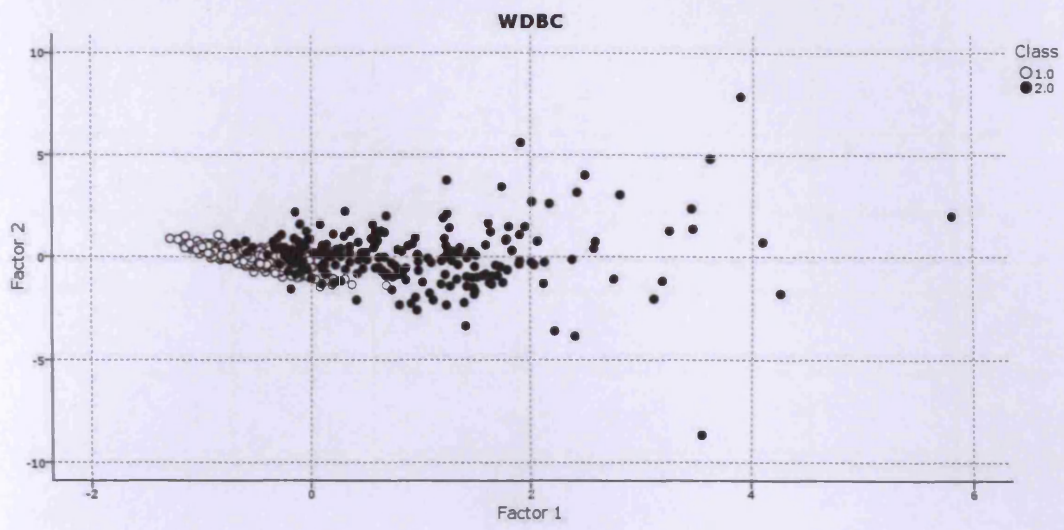


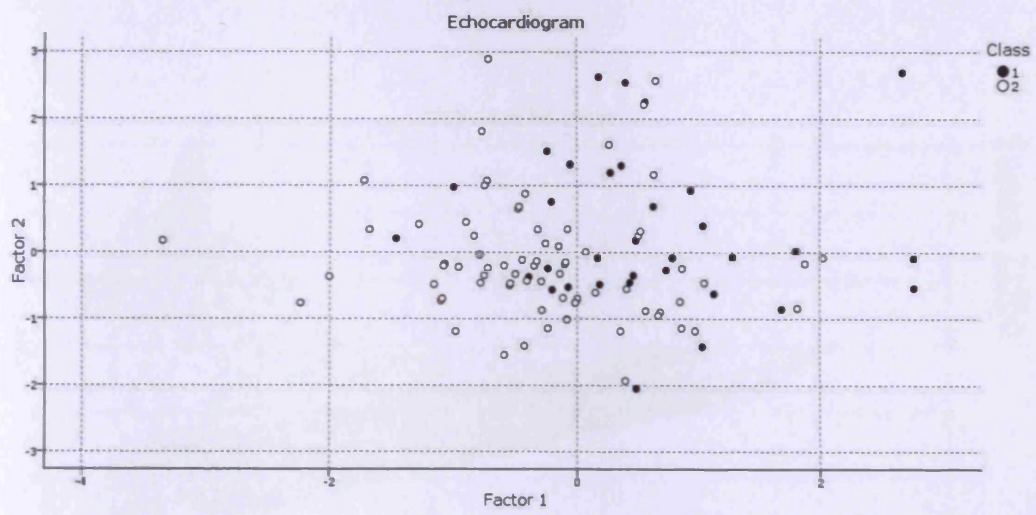
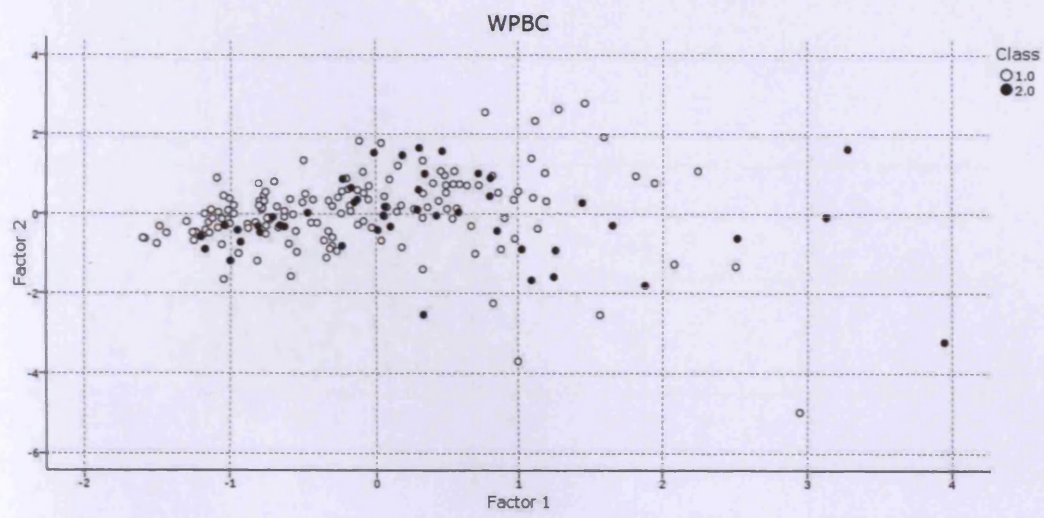


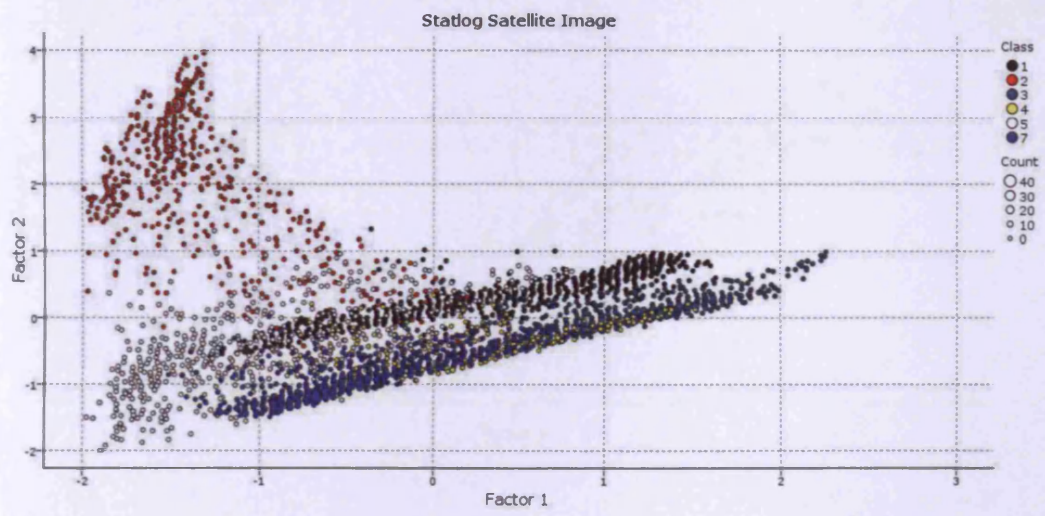
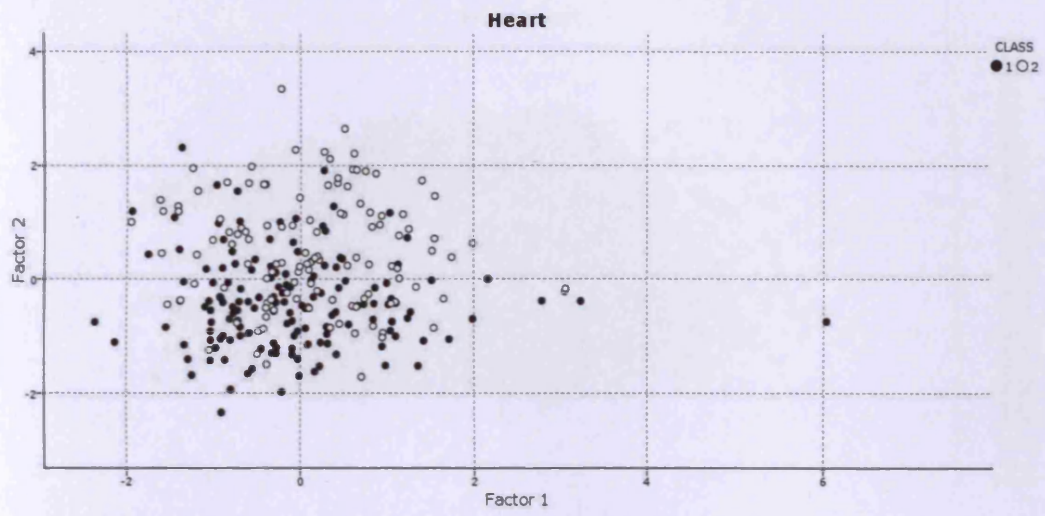


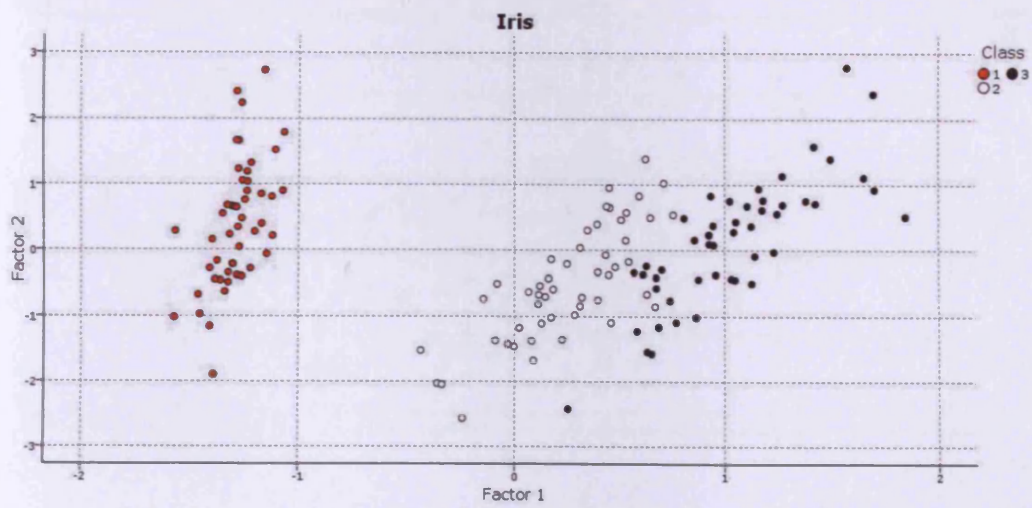
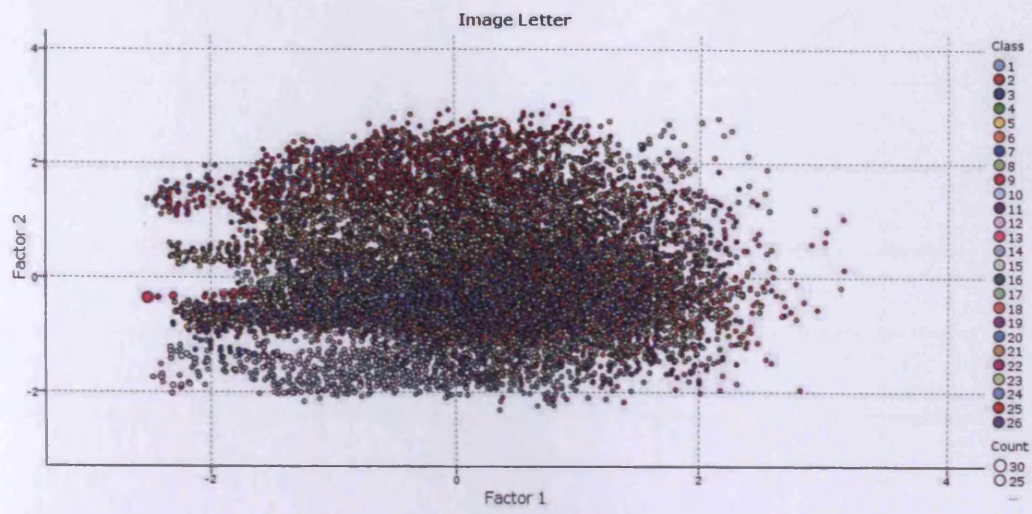


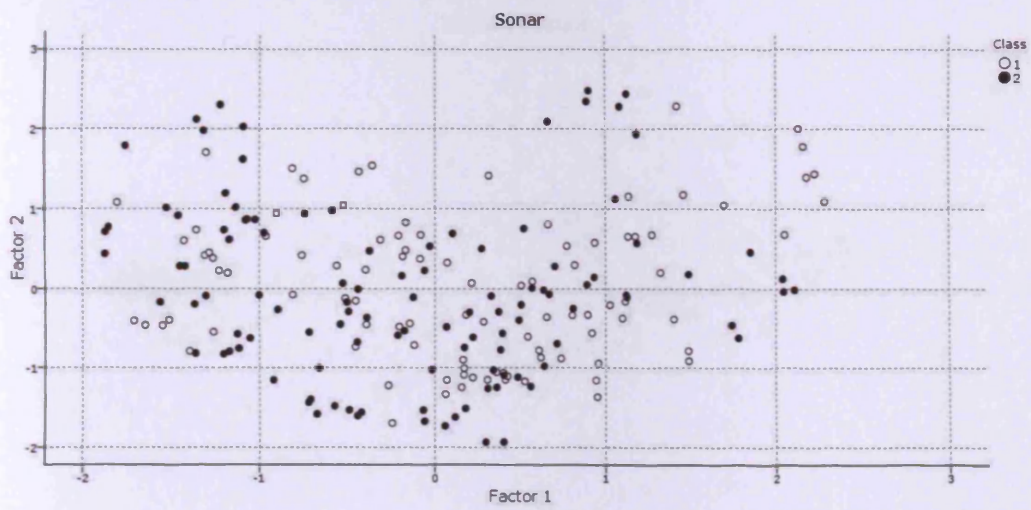
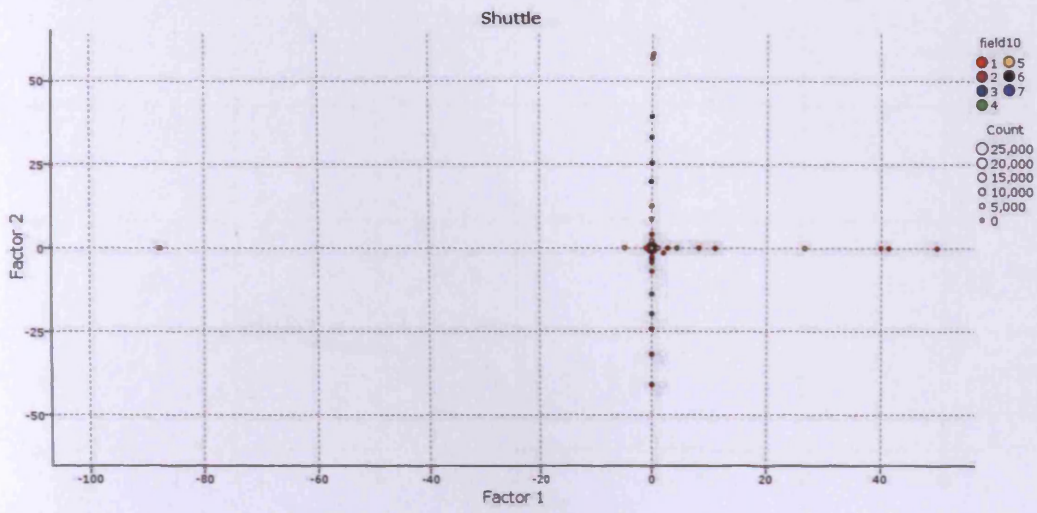


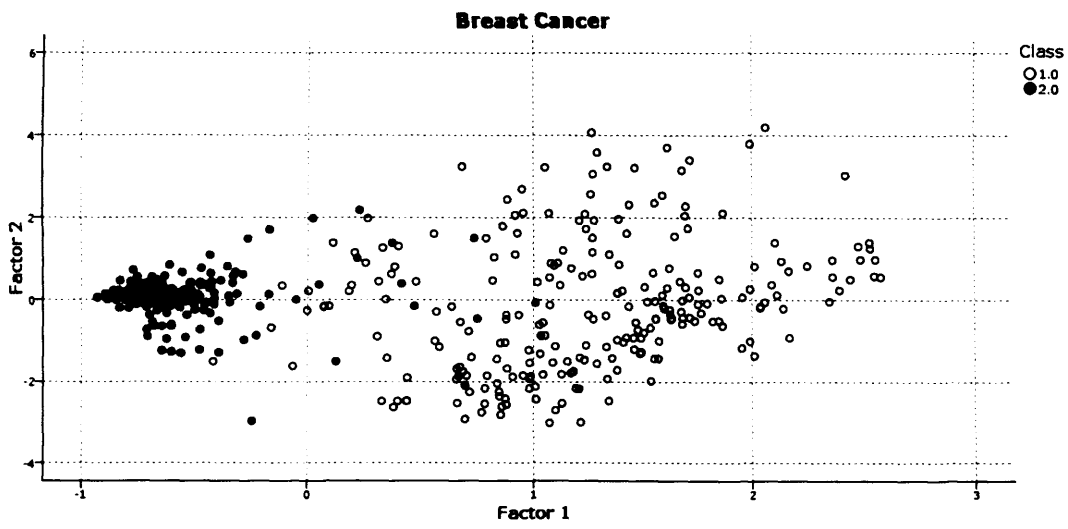
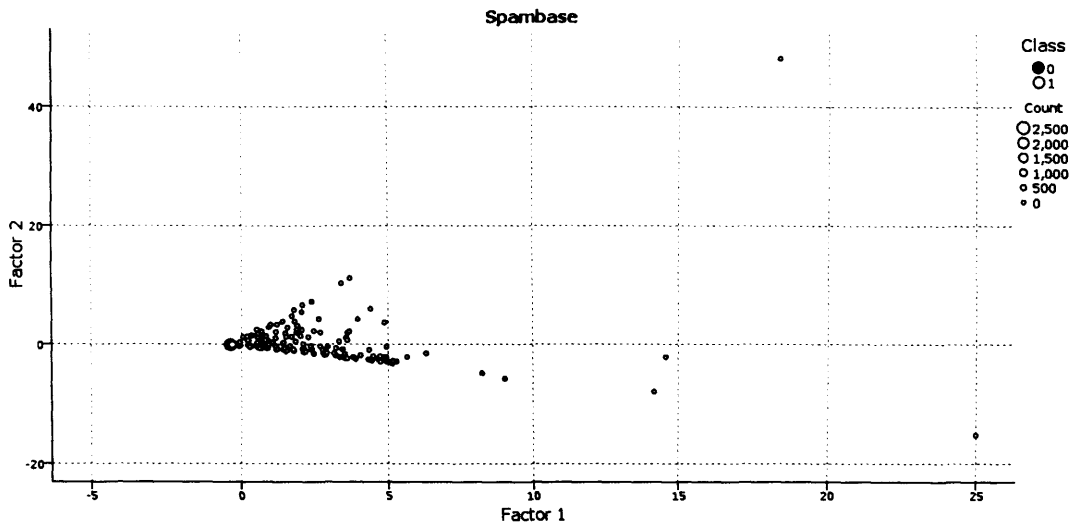




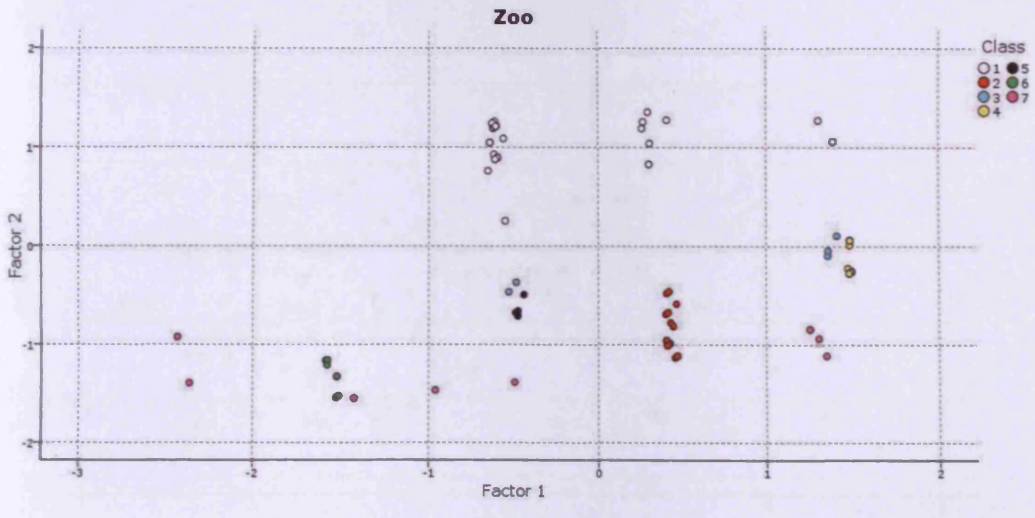
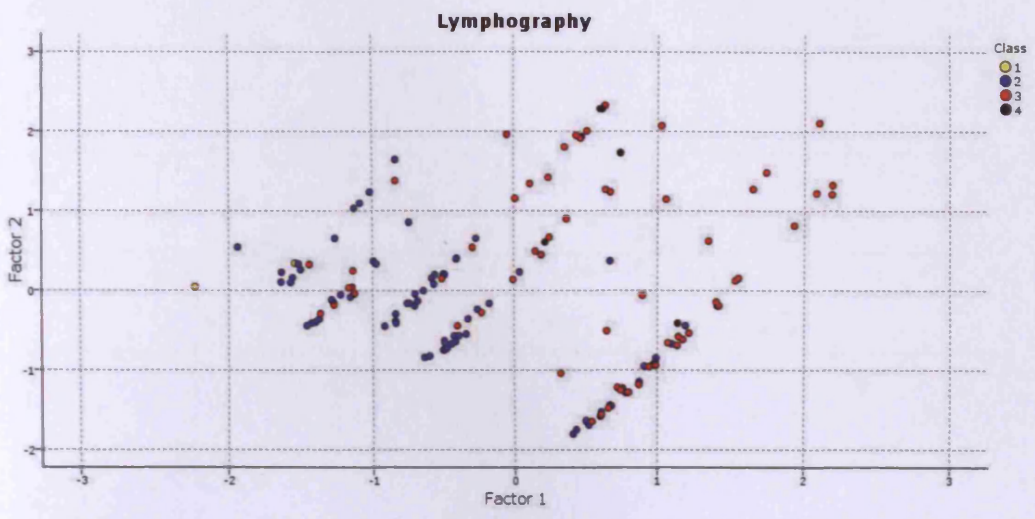


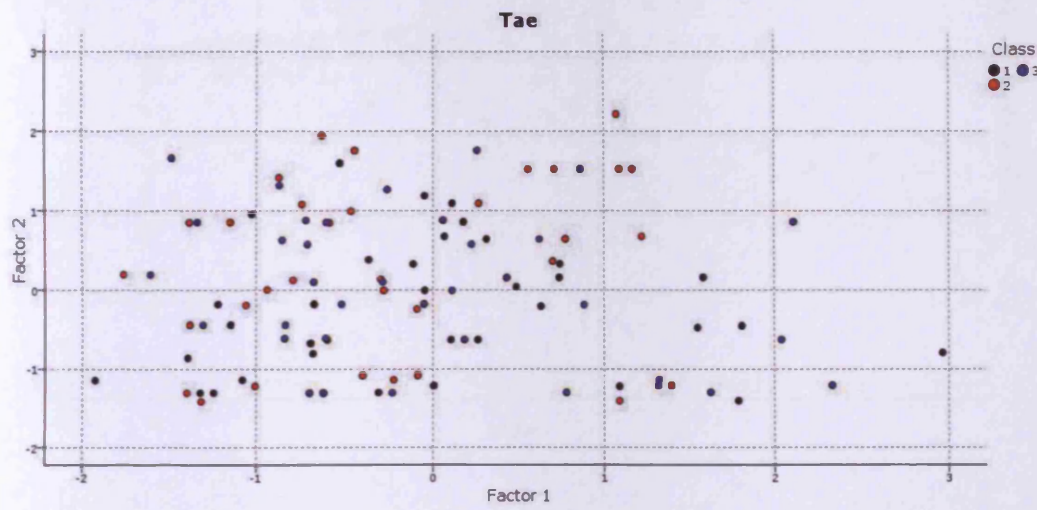
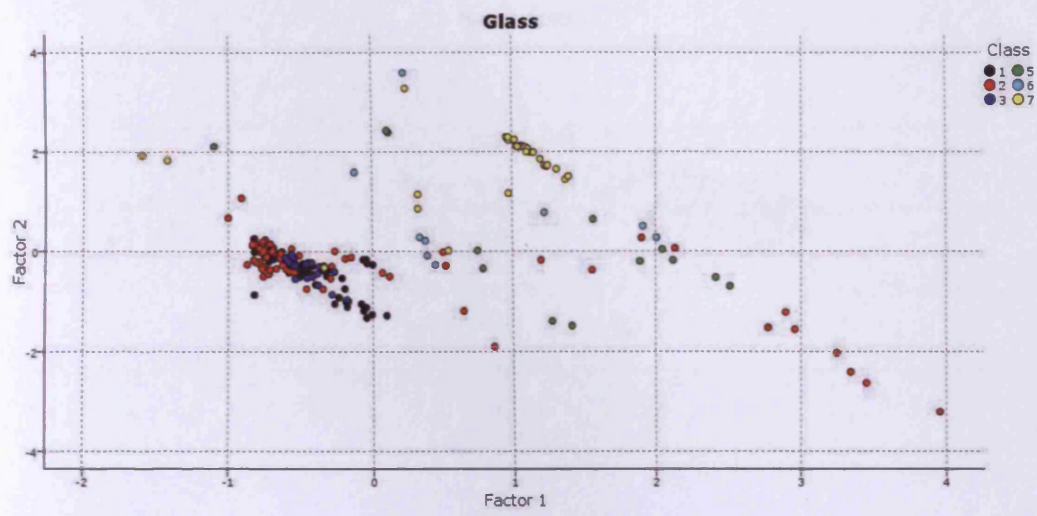


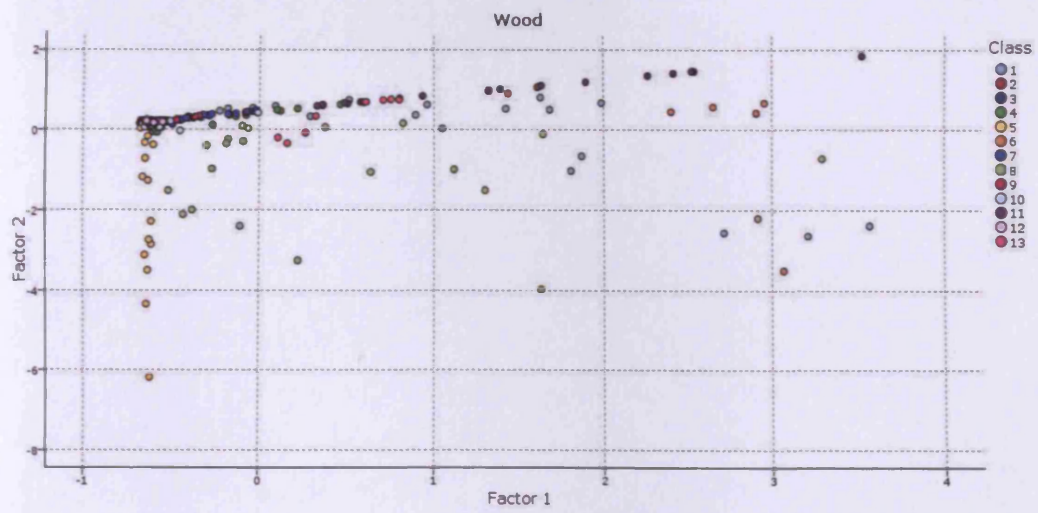
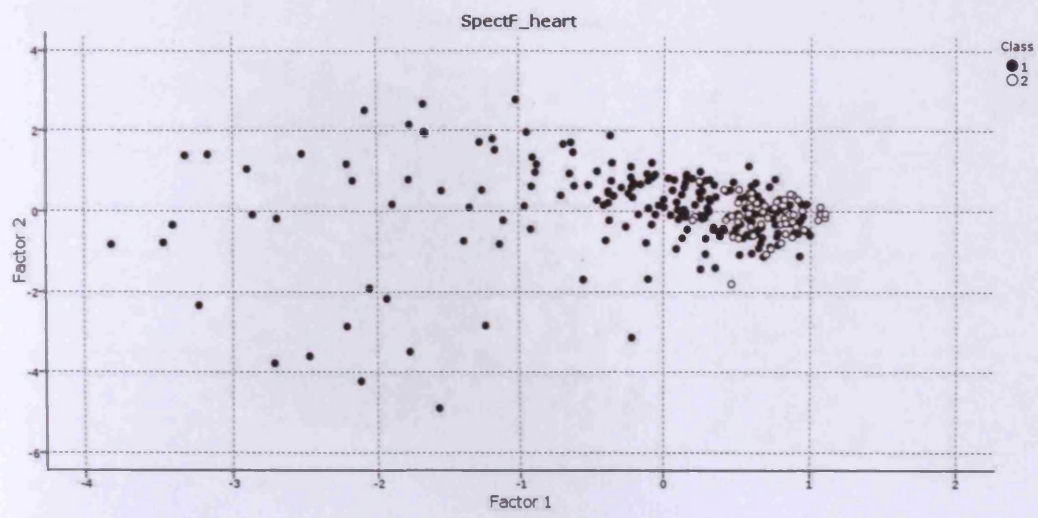


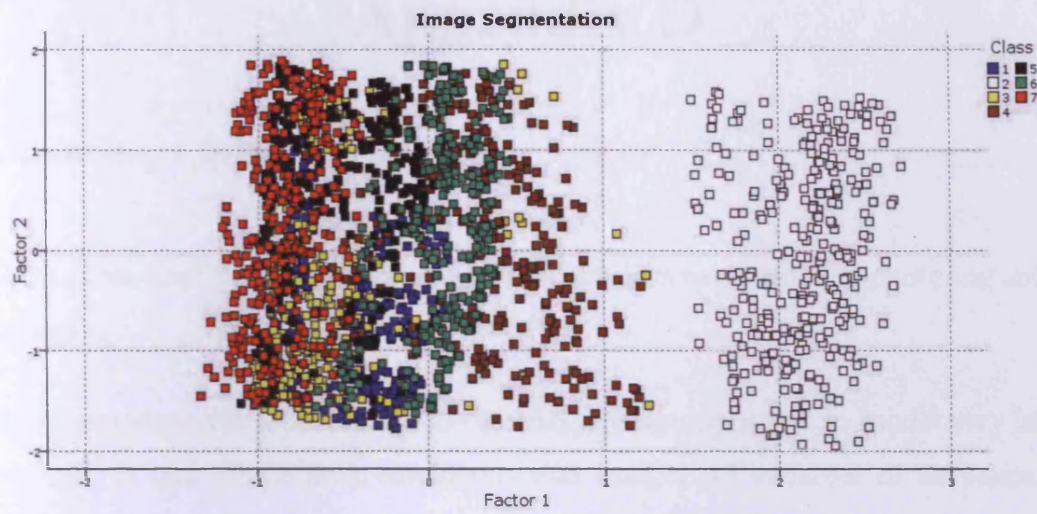












# Appendix D

## A Two Step Clustering Algorithm

As the name implies, the clustering algorithm involves two steps: Pre-clustering and Clustering.

The cluster algorithm is a scalable cluster analysis algorithm able to handle very large data sets. It can handle both continuous and categorical variables or attributes. It requires only one data pass. It has two steps 1) pre-cluster the cases (or records) into many small sub-clusters; 2) cluster the sub-clusters resulting from pre-cluster step into the desired number of clusters ISL (2007).

### First step

The pre-cluster step uses a sequential clustering approach. It scans the data records one by one and decides if the current record should be merged with the previously formed clusters or starts a new cluster based on the distance criterion (described below). The procedure works constructing a modified cluster feature (CF) tree. The CF tree consists of levels of nodes, and each node contains a number of entries. A leaf entry (an entry in the leaf node) represents a final sub-cluster. The non-leaf nodes and their entries are used to guide a new record quickly into a correct leaf node. Each entry is characterized by its CF that consists of the entry's number of records, mean and variance of each range field, and counts for each category of each symbolic field. For each successive record, starting from the root node, it is recursively guided by the closest entry in the node to find the closest child node, and descends along the CF tree. Upon reaching a leaf node, it finds the closest leaf entry in the leaf node. If the record is within a threshold distance of the closest leaf entry, it is absorbed into the leaf entry and the CF of that leaf entry is updated. Otherwise it starts its own leaf entry in the leaf node. If there is no space in the leaf node to create a new leaf entry, the leaf node is split into two. The entries in the original leaf node are divided into two groups using the farthest pair as seeds, and redistributing the remaining entries based on the closeness criterion.

If the CF tree grows beyond allowed maximum size, the CF tree is rebuilt based on the existing CF tree by increasing the threshold distance criterion. The rebuilt CF tree is smaller and hence has space for new input records. This process continues until a complete data pass is finished. For details of CF tree construction, see the BIRCH algorithm (Zhang, Ramakrishnon, and Livny,1996).

All records falling in the same entry can be collectively represented by the entry's CF. When a new record is added to an entry, the new CF can be computed from this new record and the old CF without knowing the individual records in the entry. These properties of CF make it possible to maintain only the entry CFs, rather than the sets of individual records. Hence the CF-tree is much smaller than the original data and can be stored in memory more efficiently.

Note that the structure of the constructed CF tree may depend on the input order of the cases or records. To minimize the order effect, randomly order the records before building the model.

The cluster step takes sub-clusters (non-outlier sub-clusters if outlier handling is used) resulting from the pre-cluster step as input and then groups them into the desired number of clusters. Since the number of sub-clusters is much less than the number of original records, traditional clustering methods can be used effectively. TwoStep uses an agglomerative hierarchical clustering method, because it works well with the auto-cluster method.

### **Second step**

In a second stage, the initial estimate is refined by finding the largest relative increase in distance between the two closest clusters in each hierarchical clustering stage. This is done as follows:

Starting with the model  $C_k$  indicated by the BIC criterion, take the ratio of minimum inter-cluster distance for that model and the next larger model  $C_{k+1}$ , that is, the previous model in the hierarchical clustering procedure,

$$R_2(k) = \frac{d_{\min}(C_k)}{d_{\min}(C_{k+1})}$$

where  $C_k$  is the cluster model containing  $k$  clusters and  $d_{\min}(C)$  is the minimum inter-cluster distance for cluster model  $C$ .

# References

- AGRAWAL, R. & SRIKANT, R. (1994) Fast algorithms for fast association rules in large databases *VLDB Conference* Santiago, Chile.
- AHA, D. W. & BANKERT, R. L. (1996) A comparative evaluation of sequential feature selection algorithms *Artificial Intelligence and Statistics V*. Springer-Verlag
- ALAHAKOON, D. & HALGAMUGE, S. (2000) Dynamic Self-Organizing Maps with Controlled Growth for Knowledge Discovery. *Transactions on Neural Networks* 11, 601-614.
- ARABIE, P. & HUBERT, L. (1994) Cluster Analysis in marketing research. *Advanced Methods in Marketing Research* , 160-189.
- ARYA, M., CODY, W., FALOUTSOS, C., RICHARDSON, J. & TOGA, A. (1993) A prototype 3-D medical image database system *IEEE Data Engineering Bulletin* 16, 38-42.
- AUNFFARTH, B., LOPEZ, M. & CERQUIDES, J. (2008) Hopfield Networks in Relevance and Redundancy Feature Selection Applied to classification of Biomedical High-Resolution Micro-CT Images. *find journal ICDM* 16-31.
- BARBARA, D. (2003) Using Self-Similarity to Cluster Large Data Sets. *Data Mining and Knowledge Discovery* 7, 123-152.
- BARBASI, A.-L. (2002) *Linked: The New Science of Networks* Perseus Publishing
- BARNESLEY, M. (1988) *Fractals Everywhere*, ACADEMIC PRESS INC. (LONDON) LTD.
- BATTISTI, R. (1994) Using Mutual Information for Selecting Features in Supervised Neural Net Learning. *IEEE Transactions on Neural Networks* 5.
- BAUMGARTNER, C. & PLANT, C. (2004) Subspace Selection for Clustering High-Dimensional Data. *International Conference on Data Mining*.
- BEIRLANT, E., DUDEWICZ, E., GYORFI, L. & MEULEN, E. V. D. (1996) Nonparametric Entropy Estimation *International journal on Mathematical and Statistical Science* 5, 17-39.



- BELUSSI, A. & FALOUSTOS, C. (1995) Estimating the selectivity of spatial queries using the 'Correlation Fractal Dimension'. *Proceedings of the 21st international conference on very large data bases*. Switzerland
- BHAVANI, D., RANI, T. S. & BAPI, R. (2008) Feature selection using correlation fractal dimension: issues and applications in binary classification problems. *Applied Soft Computing*, 8.
- BICHSEL, M. & SEITZ, P. (1989) Minimum class entropy: A maximum information approach to layered networks *Neural Networks*, 2, 133-141.
- BOHM, C. & KRIEGEL, H.-P. (1999).
- BORODICH, F. M. (1997) Some Fractals Models of Fracture. *Mechanics, Physics and Solids*, 45, 239-259.
- BOX, E. P. G., GWILYM, M., JENKINS, M., GREGORY, C. & REINSEL, C. (1994) Analysis: Forecasting and Control 3rd ed. Englewood Cliffs, NJ, Prentice Hall
- BRANDON, M. C., WALLACE, D. C. & BALDI, P. (2009) Data Structures and Compression Algorithms for Genomic Sequence Data. *Bioinformatics Oxford Journals*.
- BREIMAN, L., FRIEDMAN, J. H., OLSHEN, R. & STONE, C. J. (1984) *Classification and regression tree*, Pacific California, Wadsworth & Brooks/ Cole Advanced Books Software.
- CASTAGLI, M. & EUBANK, S. (1992) *Nonlinear Modeling and Forecasting*, Addison Wesley
- CHAUDHURI, B. B. & SARKAR, N. (1998) *Fractals and Beyond Complexities in the Sciences*.
- CHEN, H., SCHROEDER, J., HAUCK, R., RIDGEWAY, L., ATABAKSH, H., GUPTA, H., BOARMAN, C., RASMUSSEN, K. & CLEMENS, A. (2003) Coplink connect: Information and knowledge management for lwa enforcement *CACM*.
- CHENG, C.-H. & WEI, L.-Y. (2009) Data spread-based entropy clustering method using adaptive learning *Expert Systems with Applications*, 36, 12357-12361.
- CHIU, W. K., YEUNG, Y. C. & YU, K. M. (2006) Toolpath generation for layer manufacturing of fractal objects. *Rapid Prototyping* 12, 214-221.

- CHOW, T. & HUANG, D. (2005) Estimating Optimal Feature Subsets Using Efficient Estimation of High-Dimensional Mutual Information. *IEEE transactions on Neural Networks* 16, 213-224.
- CIOU, K. J. & LIU, N. (1992) A machine learning method for generation of a neural network architecture: A continuous ID3 algorithm *IEEE transactions in Neural Networks*, 3, 280-290.
- CONAIRE, C. O. & O'CONNOR, N. E. (2008) Unsupervised feature selection for detection using mutual information thresholding. *Ninth International Workshop on Image Analysis for Multimedia Interactive Services*. IEEE computer society.
- CORD, A., AMBROSIE, C. & COCQUEREZ, J.-P. (2005) Feature Selection in Robust Clustering based on Laplace Mixture. *Pattern Recognition Letters*, 27 627-635.
- COVER, T. M. & THOMAS, J. A. (2006) *Elements of information theory*, New york, Wiley.
- DAS, S. K. (1971) Feature Selection with Linear Dependence Measure. *IEEE Transactions in Computers*
- DASH, M., CHOI, K., SCHEUERMANN, P. & LIU, H. (2002) Feature Selection for Clustering - A Filter Solution. *Proceedings of the 2002 IEEE International Conference on Data Mining*. Maebashi City, Japan, IEEE Computer Society.
- DASH, M. & LIU, H. (2000) Feature Selection for Clustering *Knowledge Discovery and Data Mining, Current Issues and New Applications, 4th Pacific-Asia Conference. PADKK 2000* Kyoto, Japan
- DASH, M., LIU, H. & YAO, J. (1997) Dimensionality reduction for unsupervised data. *Ninth IEEE International Conference on Tools with AI*
- DEISY, C., SUBBULAKSHMI, B., BASKAR, S. & RAMARAJ, N. (2007) Efficient Dimensionality Reduction Approaches for Feature Selection. *Computational Intelligence and Multimedia Applications*. IEEE
- DIJCK, G. V. & HULLE, M. M. V. (2007) *Speeding Up Feature Subset Selection Through Mutual Information Relevance Filtering* Springer-Verlag Berlin Heidelberg
- DING, C., HE, X., ZHA, H., GU, M. & SIMON, H. (2001) A min-max cut algorithm for graph partitioning and data clustering. *IEEE International Conference in Data Mining*

- DING, C. & PENG, H. (2003) Minimum Redundancy Feature Selection from Microarray Gene Expression Data. *Proceedings fo the computational Systems Bioinformatics (CSB'03)*. Computer Society.
- DRUGMAN, T. & THIRAN, M. G. A. J.-P. (2007) Relevant Feature Selection for Audio-Visual Speech Recognition. *MMSP*.
- DUCH, W., WINIARSKI, T., BIESIADA, J. & KATCHEL, A. (2003) Feature ranking selection and discretization *Artificial Neural Networks (ICANN)*. Istanbul, Bogazici University Press
- ELLIS, D. & BILMES, J. (2000) Using mutual information to design feature combinations *ICSLP*.
- ESTER, M., KRIEGEL, H. P., SANDER, J. & XU, X. (1996) Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *2nd Int.Conf. on knowledge Discovery and Data Mining*. Portland, Oregon.
- FALOUTSOS, C., RANGANATHAN, M. & MANOLOPOULOS, Y. (1994) Fast subsequence matching in time series databases *In Proc. ACM SIGMOD* Minneapolis.
- FAST, J. D. (1962) *Entropy: The significance fo the concept of entropy ant its applications in sicence and technology*, Eindhoven: Philips Technical Library.
- FAYYAD, U., PIATETSKY-SHAPIRO, G. & SMYTH, P. (1996) From data mining to knowledge discovery in databases. *AI Magazine*, 17, 37-54.
- FERGUSON, S. T. (1973) Bayesian analysis of nonparametric problems *find journal (Ann Statist)*, 1, 209-230.
- FIGUEIREDO, M. A. T., CHANG, D. S. & MURINO, V. (2006) Clustering under prior knowledge with application to image segmentation *find journal (adv Neural Inform process Systems)*, 19, 401-408.
- FISHER, D. H. (1987) Knowledge acquisition via incremental conceptual clustering *Machine Learning* 2, 103-138.
- FLOOK, A. (1996) Features Fractals. *Sensor Review*, 16, 42-47.

- FLOREK, K., LUKASZEWICZ, J., PERKAL, J., STEINHAUS, H. & ZUBRZYCKI, S. (1951) Sur la liason et la division des points d'un ensemble fini. *fund journal Colloq Math*, 282-285.
- FRANCOIS, D., ROSSI, F., WERTZ, V. & VERLEYSEN, M. (2007) Resampling methods for parameter-free and robust feature selection with mutual information. *Neurocomputing*, 70, 1276-1288.
- FRIGUI, H. & KRISHNAPURAM, R. (1999) A robust competitive clustering algorithm with applications in computer vision. *IEEE transactions Analysis Machine Intelligence*, 21, 450-465.
- GENNARI, J., LANGLEY, P. & FISHER, D. (1989) Models of incremental concept formation. *Artificial Intelligence* 40, 11-62.
- GHEYAS, I. A. (2009) Feature subset selection in large dimensionality domains. *Pattern Recognition*, 43, 5-13.
- GHISELLI, E. E. (1964) *Theory of psychological measurement* New York.
- GOWDA, K. C. & DIDAY, E. (1991) Symbolic clustering using a new dissimilarity measure *Pattern Recognition* 24, 567-578.
- GUHA, S., MEYERSON, A., MISHRA, N. & MOTWANI, R. (2003) Clustering Data Stream: Theory and Practice. *IEEE transactions on knowledge and data engineering* 15, 515-528.
- GUYON, I. & ELISSEEFF, A. (2003) An introduction to Variable and Feature Selection *Machine Learning Research* 3, 1157-1182.
- GUYON, I., GUNN, S., NIKRAVESH, M. & ZADEH, L. A. (2006) *Feature Extraction Foundations and Applications* Springer-Verlag Berlin Heidelberg.
- GUYON, I., LI, J., MADER, T., PLETSCHER, P. A., SCHNEIDER, G. & UHR, M. (2007) Competitive baseline methods set new standards for the NIPS 2003 feature selection benchmark. *Pattern Recognition Letters* 28, 1438-1444.
- HALL, M. A. (2000) Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning *Proceedings of the Seventeenth International Conference on Machine Learning*.

- HEJAZAI, M. I. & CAI, X. (2009) Input Variable Selection for Water Resources Systems using a Modified Minimum Redundancy Maximum Relevance algorithm. *Advances in Water Resources*
- HEYDORN, R. P. (1971) Redundancy in Feature Extraction *IEEE Transactions in Computers*.
- HIROTA, K. & PEDRYCZ, W. (1985) Subjective Entropy of Probabilistic Sets and Fuzzy Cluster Analysis. *Find the original name IEEE transactions on systems Man and Cybernetics*.
- HOLTE, R. C. (1993) Very simple classification rules perform well on most commonly used dataset. *Machine Learning*, 11, 63-91.
- HONGZHAO, D., DONGGXU, L., YANWEI, Z. & YING, C. (2004) A novel approach of networked manufacturing collaboration: fractal web-based extended enterprise. *Manufacturing Technology*, 26, 1436-1442.
- HU, J., RAY, B. K. & SINGH, M. (2007) Statistical Methods for automated generation of service engagement staffing plans *find name of the journal from "general clustering" paper*, 51, 281-293.
- HULLE, M. V. & GAUTAMA, T. (2004) Optimal Smoothing of Kernel-Based Topographic Maps with Applications to Density-Based Clustering Shapes. *VLSI Signal Processing*, 37, 211-222.
- HYVARINEN, A. & OJA, E. (2000) Independent Component Analysis; Algorithms and Applications *Neural Networks* 13, 411-430.
- INZA, I., LARRANAGA, P., ETXEBERRIA, R. & SIERRA, B. (1999) Feature Subset Selection by Bayesian network-based optimization. *Artificial Intelligence*, 123, 157-184.
- IWAYAMA, M. & TOKUNAGA, T. (1995) Cluster-based text categorization: a comparison of category search strategies *18th ACM International Conference on Research and Development in Information Retrieval*
- ISL. (2007). *Clementine Data Mining Package*. SPSS UK Ltd., 1st Floor, St. Andrew's House, West Street, Woking, Surrey, United Kingdom.
- JAIN, A. K. (2009) Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31, 651-666.

- JAIN, A. K. & DUBES, R. C. (1988a) *Algorithms for Clustering Data*, Prentice Hall Englewood  
Cliff, NJ.
- JAIN, A. K. & FLYNN, P. (1996) Image Segmentation using Clustering *Advances in Image  
Understanding* 65-83.
- JAIN, K. A. & DUBES, C. R. (1988b) *Algorithms for Clustering Data*.
- JANG, R. & CHUEN-TSAI (1995) Neuro Fuzzy Modeling and Control. *Proceedings of the IEEE*,  
83.
- JING, L., MICHAEL, K. & HUANG, J. Z. (2007) An Entropy Weighting k-means Algorithm  
for Subspace Clustering of High-Dimensional Sparse Data *IEEE Transactions on Knowledge  
and Data Engineering* 19, 1026-1041.
- JOHN, G. H., KOHAVI, R. & PFLEGER, K. (1994) Irrelevant Features and the Subset  
Selection Problem. *Machine Learning: Proceedings of the Eleventh International Conference*. San  
Francisco CA, Morgan Kaufmann.
- KAILING, K., KRIEGEL, H. P., KROGER, P. & WANKA, S. (2003) Ranking Interesting  
Subspaces for Clustering High Dimensional Data *7th European Conf. on Principles and  
Practice of knowledge Discovery in Databases*.
- KARAYANNIS, N. (1994) MECA: Maximum Entropy Clustering Algorithm *3rd IEEE  
Conference on Fuzzy Systems*
- KASS, G. V. (1980) An exploratory technique for investigating large quantities of categorical  
data. *Applied Statistics* 29, 119-127.
- KAUFMAN, L. & ROUSSEEUW, P. J. (2005) *Finding Groups in Data*, Wiley & Sons.
- KEYNES, J. M. (1921) *A treatise on probability* London.
- KIRA, K. & RENDELL, L. A. (1992) The feature selection problem: Traditional methods and a  
new algorithm. *Tenth National Conference on Artificial Intelligence* MIT Press.
- KIRKPATRICK, S. & GELATT, C. D. (1983) Optimization by simulated annealing *Science* 220,  
671-680.
- KITTLER, J. (1978) Feature set search algorithms *Pattern Recognition and Signal Processing* 41-60.
- KLEINBERG, J. (2002) An impossibility theorem for clustering *find journal (NIPS)*, 463-470.

- KOHAVI, R. & JOHN, G. H. (1997a) The Wrapper Approach. *Artificial Intelligence*, 97, 273-324.
- KOHAVI, R. & JOHN, G. H. (1997b) Wrapper for feature subset selection *Artificial Intelligence*, 97, 273-324.
- KOHONEN, T. (1982) Self-Organized Formation of Topologically Correct Feature Maps *Biological Cybernetics*, 43, 59-69.
- KOLMOGOROV, A. N. (1998) On tables of random numbers. *Theoretical Computer Science*, 207, 387-395.
- KUDO, M. & SKLANSKY, J. (1999) Comparison of algorithms that select features for pattern classifiers *Pattern Recognition*, 33, 25-41.
- KUDO, M. & SKLANSKY, J. (2000) Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33, 25-41.
- LAAN, M. V. D. & POLLARD, K. (2003) A new algorithm for hybrid hierarchical clustering with visualization and the bootstrap. *Journal of statistical planning and inference*, 117, 275-303.
- LEE, H.-M., CHEN, C.-M., CHEN, J.-M. & JOU, Y.-L. (2001) An Efficient Fuzzy Classifier with Feature Selection Based on Fuzzy Entropy *check name IEEE Transactions on Systems, Manufacturing and Cybernetics* 31, 426-432.
- LEE, S. J., JONE, M. T. & SAI, H. L. (1999) Constructing Neural Networks for multi-class discretization based on information entropy. . *Find journal (Sys Man Cybern B)*, 29, 445-453.
- LI, R. & MUKAIDONO, M. (1995) A maximum entropy approach to fuzzy clustering *Conference on Fuzzy Systems*
- LI, W. (1990) Mutual Information Functions versus Correlation Functions *Journal of Statistical Physics* 60.
- LI, Y., DONG, M. & HUA, J. (2007) Localized feature selection for clustering. *Pattern Recognition Letters*, 29, 10-18.
- LIU, H., SUN, J., LIU, L. & ZHANG, H. (2008) Feature Selection with Dynamic Mutual Information *Pattern Recognition*.

- LORETTE, A., DESCOMBES, X. & ZERUBIA, J. (2000) Fully Unsupervised Fuzzy Clustering with Entropy Criterion *ICPR 00*. Barcelona.
- MACNAUGHTON, P. S., WILLIAMS, W. T., DALE, M. B. & MOCKETT, L. G. (1964) Dissimilarity analysis: A new technique of hierarchical sub-division *Nature* 202, 1034-1035.
- MALVIYA, A. (2008) Fractal Based Spatial Domain Tehcniques for Image De-Noiseing. *ICALIP*.
- MANDELBROT, B. & NESS, J. W. V. (1968) Brownian motion fractionall noises and applications *SLAM review*, 422(437).
- MANDELBROT, B. B. (1977) *Fractals, Form, Chance and Dimension*, W H FREEMAN AND COMPANY.
- MANDELBROT, B. B. (1982) *The Fractal Geometry of Nature*, W H FREEMAN AND COMPANY.
- MARILL, T. & GREEN, D. M. (1963) On the effectiveness of receptors in recognition systems *IEEE transactions on information theory*, 9, 11-17.
- MATLAB <http://www.mathworks.com/matlabcentral/>.
- MICHALEWICZ, Z. (1992) *Genetic Algorithms + Data Structures = Evolution Programs*, Springer Verlag
- MICHEL, V., DAMON, C. & THIRION, B. (2008) Mutual Information-Based Feature Selection Enhances FMRI Brain Activity Classification. IEEE Xplore.
- NANNI, M. (2005) Speeding -Up Hierarchical Agglomerative Clustering in Presence of Expensive Metrics *Pacific Asia Knowledge Discovery and Data Mining Conferences* 378-387.
- NARENDRA, P. M. & FUKUNAGA, K. (1997) A branch and bound algorithm for features subset selection. *IEEE transactions on computers*, 29, 917-922.
- OGURA, H., AMANO, H. & KONDO, M. (2010) Distinctive Characteristics of a metric using deviations from Possion for feature selection. *Expert Systems with Applications*, 37, 2273-2281.
- PAPADIMITRIOU, S., BROCKWELL, A. & FALOUSTSOS, C. (2003) Adaptive, hands-off stream mining *VLDB*.



- PAVAN, M. & PELILLO, M. (2003) New Graph-Theoretic Approach to Clustering and Segmentation. *Proceedings of the 2003 IEEE Computer Society Conference on computer Vision and Pattern Recognition*, 1063-6919.
- PENG, H., LONG, F. & DING, C. (2005) Feature Selection Based on Mutual information: Criteria of Max-Dependence, Max-Relevance, and Min-Redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, 27.
- PONSA, D. & LOPEZ, A. (2007) Feature Selection Based on a New Formulation of the Minimal-Redundancy-Maximal-Relevance Criterion. Springer-Verlag Berlin Heidelberg.
- PRASAD, M. G. P., DUBE, S. & SRIDHARAN, K. (2003) An efficient fractal-based algorithm for clustering *IEEE Region 10 Conference. Convergent Technologies for the Asia-Pacific*
- PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T. & FLANNERY, B. P. (1988) *Numerical recipes in C. The art of scientific computing*, Cambridge University Press.
- PUDIL, P., NOVOVICOVA, J. & KITTLER, J. (1994) Floating search methods in feature selection. *Pattern Recognition Letters*, 15, 1119-1125.
- PUDIL, P., NOVOVICOVA, J. & SOMOL, P. (2002) Feature Selection toolbox software package *Pattern Recognition Letters*, 23, 487-492.
- QUINLAN, J., R. (1995) *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA.
- QUINLAN, J. R. (1986) Induction of Decision Trees *Machine Learning*, 1.
- RASMUSSEN, C. E., CRUZ, B. J. D. L., GHAHRAMANI, Z. & WILD, D. L. (2009) Modeling and Visualizing Uncertainty in Gene Expression Clusters Using Dirichlet Process Mixtures. *IEEE /ACM Transactions on Computational Biology and Bioinformatics*, 6, 615-626.
- REYNOLDS, H. T. (1977) *The analysis of cross-classifications*, The Free Press , NY USA.
- ROBNIK-SIKONJA, M. & KONONENKO, I. (2003) Theoretical and Empirical Analysis of Relief and RRelief. *Machine Learning* 53, 23-69.
- RODRIGUES, P. P., GAMMA, J. A. & PEDROSO, J. A. P. (2007) Hierarchical Clustering of Time Data Streams. *IEEE transactions on knowledge and data engineering*, 10.

- SANCHEZ, L., SUAREZ, M. R., VILLAR, J. R. & COUSO, I. (2007) Mutual information-based feature selection and partition design in fuzzy rule-based classifiers from vague data. *International Journal of Approximate Reasoning*
- SCHROEDER, M. (1991) *Chaos, Power Laws: Minutes from an infinite Paradise*, New York, W H Freeman.
- SCOTT, D. (1992) *Multivariate Density Estimation: Theory, Practice and Visualization* New York, Wiley.
- SHI, JIANBO, MALIK & JITENDRA (2000) Normalize cuts and image segmentation. *IEEE transactions Analysis Machine Intelligence*, 22, 888-905.
- SIEDLECKI, W. & SKLANSKY, J. (1988) On automatic feature selection. *Pattern Recognition and Artificial Intelligence*, 197-220.
- SOMOL, P., PUDIL, P. & NOVOVICOVA, J. (1999) Adaptive floating search methods in feature selections. *Pattern Recognition Letters*, 20, 1157-1163.
- STRACUZZI, D. J. & UTGOFF, P. E. (2004) Randomized variable elimination *Journal of Machine Learning Research*, 5, 1331-1362.
- TALAVERA, L. (2005) An Evaluation of Filters and Wrappers Methods for Feature Selection in Categorical Clustering. 440-451.
- TASOULIS, D. K. & VRAHATIS, M. N. (2005) Unsupervised Clustering using Fractal Dimension. *International Journal of Bifurcation and Chaos* 16, 2073-2079.
- TIBSHIRANI, R., WALTER, G. & HASTIE, T. (2001) Estimating the number of clusters in a data set via the gap statistic (*find journal J roy Statist Soc B*), 411-423.
- TOUSSANI, G. T. & VILMANSEN, T. R. (1972) Comments on Feature Selection with a Linear Dependence Measure. *IEEE Transactions in Computers*
- TRAINA, C., JR., TRAINA, A., WU, L. & FALOUSOS, C. (2000) Fast feature selection using fractal dimension.
- TSAI, H.-L. & LEE, S.-J. (2004) Entropy-Based Generation of Supervised Neural Networks for Classification of Structured Patterns. *IEEE transactions on Neural Networks*, 15.

- VIOLA, P. (1995) Alignment by Maximization of Mutual Information *5th International Conference in Computer Vision* Los Alamitos.
- VIOLA, P., SCHRAUDOLPH, N. N. & SEJNOWSKI, T. J. (1996) Empirical Entropy Manipulation for Real-World Problems *find Journal (Adv in Neural Infor Proces Systems)*, 8.
- WANG, C., KNIGHT, J. C. & ELDER, M. C. (2000) On computer viral infection and the effect of immunization *ACSAC*
- WANG, L.-Y. (2005) Fractal Clustering and Knowledge-driven Validation Assesment for Gene Expression Profiling. *Engineering in Medicine nad biology 27th Annual Conference* Shanghai, China
- WANG, S. J. & SONTAKKE, T. R. (2005) A Scalable Approach for Packet Classification Using Rule-Based Partition. *CNIR*, 5, 19-26.
- WHITNEY, W. (1971) A direct method of nonparametric measurements selection. *IEEE transactions on computers*, 20, 1100-1103.
- XIONG, H., STEINBACH, M., RUSLIM, A. & KUMAR, V. (2009a) Characterizing pattern preserving clustering *Knowledge Information Systems* 19, 311-336.
- XIONG, T., WANG, S., MAYERS, A. & MONGA, E. (2009b) A New MCA-based Divisive Hierarchical Algorithm for Clustering Categorical Data. *Ninth IEEE International Conference on Data Mining*
- YAO, J., DASH, M., TAN, S. T. & LIU, H. (1998) Entropy-based fuzzy clustering and fuzzy modeling. *Fuzzy sets and systems*, 113, 381-388.
- YU, L. & LIU, H. (2004) Efficient Feature Selection via Analysis of Relevance and Redundancy. *Machine Learning Research*, 5, 1205-1224.
- ZENG, H. & CHEUNG, Y.-M. (2009) A new Feature Selection Method for Gaussian Mixture Clustering. *Pattern Recognition*, 42, 243-250.
- ZHANG, T., RAMAKRISHNAN, R. & LIVNY, R. (1996) BIRCH: an efficient data clustering model for very large databases. 1996 ACM SIGMOD international conference on management of data.

ZHAO, L., MA, N. & XU, X. (2008) Face Recognition Based on fractal Dimension *World Congress on Intelligent Control and Automation*. Chongqing, China.

ZHOU, W., ZHOU, C., ZHU, H., LIU, G. & CHANG, X. (2006) Feature Selection for Microarray Data Analysis Using Mutual Information and Rough Set Theory. *ICIC*

ZHU, S., WANG, D., YU, K., LI, T. & GONG, Y. (2010) Feature Selection for Gene Expression Using Model-Based Entropy *IEEE /ACM Transactions on Computational Biology and Bioinformatics*, 7, 26-36.

ZHUANG, X. & MENG, Q. (2004) Local fuzzy fractal dimension and its applications in medical image processing. *Artificial Intelligence in Medicine*, 32, 29-36.

