Bayesian networks for classification, clustering, and high-dimensional data visualisation

A thesis

submitted to Cardiff University

for the degree of

Doctor of Philosophy

by

Gonzalo Andrés Ruz Heredia

Manufacturing Engineering Centre

Cardiff University

United Kingdom

May 2008

SUMMARY

This thesis presents new developments for a particular class of Bayesian networks which are limited in the number of parent nodes that each node in the network can have. This restriction yields structures which have low complexity (number of edges), thus enabling the formulation of optimal learning algorithms for Bayesian networks from data. The new developments are focused on three topics: classification, clustering, and high-dimensional data visualisation (topographic map formation).

For classification purposes, a new learning algorithm for Bayesian networks is introduced which generates simple Bayesian network classifiers. This approach creates a completely new class of networks which previously was limited mostly to two well known models, the *naive Bayesian* (NB) classifier and the *Tree Augmented Naive Bayes* (TAN) classifier. The proposed learning algorithm enhances the NB model by adding a Bayesian monitoring system. Therefore, the complexity of the resulting network is determined according to the input data yielding structures which model the data distribution in a more realistic way which improves the classification performance.

Research on Bayesian networks for clustering has not been as popular as for classification tasks. A new unsupervised learning algorithm for three types of Bayesian network classifiers, which enables them to carry out clustering tasks, is introduced. The resulting models can perform cluster assignments in a probabilistic way using the posterior probability of a data point belonging to one of the clusters. A key characteristic of the proposed clustering models, which traditional clustering techniques do not have, is the ability to show the probabilistic dependencies amongst the variables for each cluster. This feature enables a better understanding of each cluster.

The final part of this thesis introduces one of the first developments for Bayesian networks to perform topographic mapping. A new unsupervised learning algorithm for the NB model is presented which enables the projection of high-dimensional data into a two-dimensional space for visualisation purposes. The Bayesian network formalism of the model allows the learning algorithm to generate a density model of the input data and the presence of a cost function to monitor the convergence during the training process. These important features are limitations which other mapping techniques have and which have been overcome in this research.

to my mum and dad.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor Professor D.T. Pham for his excellent guidance throughout my research, as well as his practical and positive thinking which was very useful during the more difficult stages of my research.

I would also like to thank my family for the support and encouragement they have given me always, and my wonderful girlfriend Pam for her support and unconditional love.

Special thanks to my friend Pedro Ortega who introduced me to the field of Bayesian networks several years ago and with whom I have always been able to have fruitful discussion regarding my work.

Thank you to the members of the MEC Machine Learning Group for providing interesting comments and presentations in our weekly meetings.

Thank you to the people in the computer lab where I work, for making it a cheerful place to study, as well as everyone at the MEC.

Finally I would like to thank the ORS award and the MEC for funding my research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

X_i	The *i-th* attribute.
$\mathbf{X}$	A set of attributes.
$P(\cdot)$	Probability distribution.
$p(\cdot)$	Probability density function.
e	Number of edges in the Bayesian network.
$\mathbf{E}$	A set of edges.
G	A graph.
Π_{X_i}	The set of parents of X_i in the Bayesian network.
n	Number of attributes.
N	Number of instances (examples) in the data set.
$l(\cdot)$	Codelength.
C	The class label.
$I(X;Y)$	Mutual information.
$I(X;Y \mid Z)$	Conditional mutual information.
$\mathbf{D}$	The data set.
β^{-1}	Normalising factor in the Bayes rule.
z_k^r	Indicator vector for the *r-th* example and the *k-th* mixture component.
θ	A parameter of a Bayesian network.
P	A set containing partitions of the N data instances.
P_k	The *k-th* partition.
n_k	The number of examples in P_k.

$LL(\cdot)$	Log-likelihood.
$\hat{P}(\cdot)$	Empirical distribution.
$\hat{I}(\cdot)$	Mutual information and Conditional mutual information computed by empirical distributions.
$t_k^m(\mathbf{x}^r)$	Posterior probability for the *r-th* data example in the *m-th* iteration for the *k-th* mixture component.
$\mathbf{y}_k$	Coordinate position in the output space of the *k-th* neuron.
ω_{ik}	The mean of the *i-th* attribute and the *k-th* neuron.
σ_{ik}	The variance of the *i-th* attribute and the *k-th* neuron.
$\gamma_k^{(j+1)}$	Posterior probability for the *k-th* neuron in the $j+1$ iteration.
π_k	Prior probability for the *k-th* neuron.
k^*	Winning neuron.
Λ	Neighbourhood function.
$\sigma(t)$	Neighbourhood function range at time t.
$\alpha(e)$	Learning rate at epoch e.
$M_1(k)$	Trustworthiness measure for the k closest neighbours.
$M_2(k)$	Continuity measure for the k closest neighbours.

ABBREVIATIONS

AVI	Automatic Visual Inspection
BC	Bound and Collapse
BIC	Bayesian Information Criterion
CEM	Classification Expectation Maximisation
CL	Chow and Liu multinets
CML	Classification Maximum Likelihood
CS	Cheeseman-Stutz
CTP	Conditional Probability Table
DAG	Directed Acyclic Graph
EM	Expectation Maximisation
GTM	Generative Topographic Mapping
KL	Kullback-Leibler divergence
LL	Log-Likelihood
MA	Model Averaging
MAP	Maximum *a Posteriori*
MDL	Minimum Description Length
ML	Maximum Likelihood
MLP	Multi-Layer Perceptron
MWST	Maximum Weighted Spanning Tree
NB	Naive Bayesian (or Bayes)
NBM	Naive Bayes Mapping
PCA	Principal Component Analysis

RBF	Radial Basis Function
SBN	Simple Bayesian Network classifier
SOM	Sclf-Organising Maps
TAN	Tree Augmented Naive Bayes

Chapter 1

INTRODUCTION

This chapter introduces the motivation and objectives of the research, and the methods adopted. The chapter also outlines the general structure of the thesis.

1.1 Motivation

Bayesian networks or Bayesian Belief Networks (Pearl, 1988) encode the joint probability distribution of a finite set of discrete random variables as a directed acyclic graph. Their main application originally was for knowledge representation under conditions of uncertainty and they became very popular in the medical sciences in applications such as diagnosis of disease, selection of optimal treatment alternatives, and prediction of treatment outcome in different areas (Lucas *et al.*, 2004). At the beginning, most of the Bayesian networks were constructed by hand by a human expert in the domain of the application. Then a major breakthrough occurred when researchers from the *Machine Learning* area developed algorithms for learning Bayesian networks from data (Heckerman, 1995; Heckerman *et al.*, 1995). Nevertheless, in general, constructing Bayesian networks from data is a very difficult problem. In fact, it was found that inference in a Bayesian network is NP-hard

(Cooper, 1990). For data mining applications such as classification and clustering it was found that complex structures for Bayesian networks are not needed and that a very basic model called the *naive Bayes* (NB) classifier (Duda and Hart, 1973) obtains surprisingly good results considering its simplicity. An extension of this simple model, called the *Tree Augmented Naive Bayes* (TAN) classifier (Friedman *et al.*, 1997), was developed to overcome some of the naive Bayes limitations. Many other Bayesian network classifiers have been proposed but none of them are as mathematically elegant as the TAN model.

In the context of data mining applications, the first motivation for the research presented in this thesis was to develop a Bayesian network classifier capable of augmenting (adding edges to) the naive Bayes model in a more robust and efficient way than fitting the training data to a tree structure like the TAN model. Second, in the case of clustering tasks, Bayesian networks have not had that many developments as for classification, most of the time the naive Bayes model is used for clustering because of its simplicity but not much work has been done for more complex structures like the TAN model. Third, another important application for data mining is high-dimensional data visualisation. In this field, Bayesian networks have hardly been investigated. Therefore, it is interesting to explore if this kind of task can be performed by a Bayesian network and how this can be achieved. The final motivation for this research is to encourage the use of Bayesian networks for data mining as an alternative to more popular computational intelligence techniques such as artificial neural networks, fuzzy logic and inductive learning algorithms.

1.2 Aim and Objectives

The aim of this research is to develop new learning (training) algorithms for Bayesian networks for classification, clustering and high-dimensional data visualisation.

The specific objectives are:

1. To develop a Bayesian network classifier with a limited structure (one parent per node at the most) capable of augmenting the naive Bayes model. The resulting classifier model will yield structures that are not necessarily tree structures like the TAN model, thus not forcing the data to be modelled by a predefined structure. The learning of the proposed model should follow a mathematical framework such as cross-entropy (Kullback-Leibler divergence) minimisation or log-likelihood maximisation.

2. To develop an unsupervised learning algorithm to train Bayesian networks for clustering tasks. The algorithm should efficiently train tree structure models like TAN or the previously proposed model for classification under a mathematical framework like maximum likelihood. Cluster assignments should be obtained through the posterior probabilities instead of a distance metric, most common in traditional clustering algorithms like the *k-means*.

3. To develop an unsupervised learning algorithm to train Bayesian networks for topographic map formation. Because no major work has been done in this area, the feasibility of the simple Bayesian network, naive Bayes model, will be explored. The proposed mapping technique should overcome some of the

limitations that existing mapping technique have, like the limitations of the *Self Organising Maps*.

The expected results for the proposed classifier is that it should obtain better classification performances than the TAN model and the naive Bayes model, or at least perform just as well for some cases. This is because of the robust structure learning employed in the training process which allows more realistic modelling of the training data set than the other two traditional methods. For the proposed unsupervised training method the expected outcome is that it should perform better than traditional methods such as the k-means algorithm. This is because the probabilistic model does not use a distance metric to compute the "similarity" or "closeness" of a data point to a cluster but instead it employs a posterior probability to compute cluster membership values, making the proposed method capable of handling many types of data distributions. The final research on topographic map formation using Bayesian networks should prove that topology preserving maps can be formed using this approach. Also, because of the probabilistic nature of the model, the proposed method should overcome the limitations of traditional techniques, such as the self-organising maps (SOM) which have problems including the lack of a density model and a cost function to monitor convergence.

1.3 Methods

For the three topics analysed in this thesis, each one will follow the same problem solving approach to reach the desired objectives. The methods used in this research are summarised as follows:

- *Literature review*: the most relevant papers for each research topic will be reviewed pointing out the key results, advantages and disadvantages. This should help to lay the groundwork for the research.

- *Probabilistic framework*: Bayesian networks provide a mathematically sound formalism and that property should be present in the new developments. To achieve this, the proposed methods will be developed under a maximum likelihood framework.

- *Experiments*: Although the proposed algorithms will be developed under a probabilistic framework, practical experiments are also required to see if the new developments really work. Each proposed method will be tested using machine learning benchmark data sets as well as industrial application data, like the "wood defect data set". In each case, performance measures will be computed to assess the effectiveness of the new methods and comparisons with traditional methods will be carried out.

1.4 Outline of the thesis

The thesis is organised in six chapters. The topics addressed in each chapter are as follows:

Chapter 2: In this chapter the notation and brief concepts about probabilities, graph theory, and the Markov condition are introduced. Then Bayesian networks are defined and also comments on learning Bayesian networks are given. The chapter ends with a literature review of the most recent applications of Bayesian networks in different areas.

Chapter 3: An efficient learning method for building Bayesian network classifiers is presented. The new method augments the naive Bayesian classifier using the Chow and Liu tree construction method, but it introduces a Bayesian approach to control the accuracy and complexity of the resulting network, yielding structures that are not necessarily a spanning tree. It is shown that the procedure used to construct the network minimises the cross-entropy, maximises the likelihood, and minimises the upper bound of the Bayes probability of error.

Chapter 4: In this chapter, a new approach to the unsupervised training of Bayesian network classifiers is presented. Three models have been analysed: the Chow and Liu multinets, the Tree Augmented Naive Bayes, and the new Bayesian network classifier proposed in the previous chapter which is more robust in its structure learning. In order to perform the unsupervised training of these models, the classification maximum likelihood criterion is used. The proposed method is capable of learning the Bayesian network classifiers, structure, and parameters, as well as identifying cluster labels for each example of the data set used, which enables these models to carry out clustering tasks.

Chapter 5: This chapter explores the possibility of using a simple Bayesian network model for high-dimensional data visualisation. To achieve this, a new learning algorithm is proposed in order for the naive Bayes model to perform topographic mapping. The training is carried out under the maximum likelihood framework, by means of an on-line Expectation Maximisation algorithm with a self-organising principle.

Chapter 2

BACKGROUND

This chapter presents the notation as well as basic concepts of probability theory related to the Bayesian network formalism. The Bayesian network is defined and the learning process for Bayesian networks is discussed. Finally, the most recent applications of Bayesian networks in different areas are reviewed.

2.1 Random variables and conditional independence

Throughout this thesis the notation used in Friedman *et al.* (1997) will be employed. The notation is as follows: capital letters X,Y,Z are used to denote random variable names and lower-case letters, x, y, z, to denote the specific values that those variables take. Also, sets of variables will be represented by boldface capital letters, $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ and assignments of values to the variables in the sets are denoted by boldface lower-case letters, $\mathbf{x}, \mathbf{y}, \mathbf{z}$. Any change from this standard notation will be clearly stated in the thesis.

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

A *random variable* X has a state-space Ω_X consisting of the possible values x. The state-space of a vector of random variables **X** is the Cartesian product of the individual state-spaces of the variables X_i in **X**, i.e., $\Omega_{\mathbf{X}} = \prod_i \times \Omega_{X_i}$.

A *joint probability distribution* $P(\mathbf{X})$ over **X** is a mapping $\Omega_{\mathbf{X}} \to [0;1]$ with $\sum_{\mathbf{x}} P(\mathbf{x}) = 1$. The *marginal distribution* over **Y** where $\mathbf{X} = \mathbf{Y} \cup \mathbf{Z}$ is $P(\mathbf{Y}) = \sum_{\mathbf{z}} P(\mathbf{z}, \mathbf{Y})$. Also, it is said that **Y** is *independent* of **Z** if $P(\mathbf{Y},\mathbf{Z}) = P(\mathbf{Y})P(\mathbf{Z})$. The *conditional probability distribution* $P(\mathbf{Z} \mid \mathbf{y})$ is defines as $P(\mathbf{Z},\mathbf{y}) / P(\mathbf{y})$ for $P(\mathbf{y}) > 0$.

Conditional independence between random variables is a measure of irrelevance between variables that comes about when the values of some other variables are given. Formally, it is said that **X** and **Y** are *conditionally independent* given **Z**, if $P(\mathbf{X} \mid \mathbf{Z},\mathbf{Y}) = P(\mathbf{X} \mid \mathbf{Z})$ whenever $P(\mathbf{Y},\mathbf{Z}) > 0$.

The above equation can be expressed using the notation in Neapolitan (2004), then the statement **X** and **Y** are conditionally independent given **Z** can be written as $I_P(\mathbf{X},\mathbf{Y} \mid \mathbf{Z})$.

2.2 Graph theory

A *graph* is a pair (**X**, **E**), where $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ is a set of *n nodes* (or *vertices*), and **E** is a set of ordered pairs $\mathbf{X} \times \mathbf{X}$. Elements of **E** are called *edges* (or *arcs*), if $(X_i, X_j) \in \mathbf{E}$ and $(X_j, X_i) \in \mathbf{E}$ then node X_i and X_j are connected by an *undirected* *edge*, and it can be written as $X_i - X_j$. When $(X_i, X_j) \in \mathbf{E}$ and $(X_j, X_i) \notin \mathbf{E}$ then X_i and X_j are connected by a *directed edge*, and it can be written as $X_i \rightarrow X_j$. If there is $X_i \rightarrow X_j$ or $X_j \rightarrow X_i$, or $X_i - X_j$, it is said that X_i and X_j are *adjacent*.

A graph where all edges are directed is called a *directed graph*, and a graph without directed edges is called an *undirected graph*. For a set of nodes $\{X_1, X_2, \ldots, X_n\}$, where $n \geq 2$, such $(X_{i-1}, X_i) \in \mathbf{E}$ for $2 \leq i \leq n$. Then, the set of edges connecting the n nodes is called a *path* from X_1 to X_n. The nodes $X_2, \ldots, X_{n-1}$ are called *interior nodes* on path $\{X_1, X_2, \ldots, X_n\}$. The *subpath* of path $\{X_1, X_2, \ldots, X_n\}$ from X_i to X_j is the path $\{X_i, X_{i+1}, \ldots, X_j\}$ where $1 \leq i < j \leq n$. A *directed cycle* is a path from a node to itself. A *simple path* is a path containing no subpaths which are directed cycles.

A directed graph G is called a *directed acyclic graph* (DAG) if it contains no directed cycles. Given a DAG $G = (\mathbf{X}, \mathbf{E})$ and nodes X_i and X_j in $\mathbf{X}$, when $X_j \rightarrow X_i$ then X_j is called a *parent* of X_i, and X_i is called a *child* of X_j. The node X_j is called a *descendent* of X_i and X_i is called an *ancestor* of X_j if there is a path from X_i to X_j, and X_j is called a *nondescendent* of X_i if X_j is not a descendent of X_i.

2.3 The Markov Condition

Let P be a joint probability distribution of the random variables in some set $\mathbf{X}$ and $G = (\mathbf{X}, \mathbf{E})$ a DAG. It is said that (G, P) satisfies the *Markov condition* if for each variable $X_i \in \mathbf{X}$, X_i is conditionally independent of the set of all its nondescendents given the set of all its parents. By adopting the following notation for the set of parents of X_i, $\mathbf{X}_{pa(i)}$, and the set of nondescendents of X_i, $\mathbf{X}_{nd(i)}$, then the above definition means that

$$I_P(X_i, \mathbf{X}_{nd(i)} \mid \mathbf{X}_{pa(i)}).$$

When (G, P) satisfies the Markov condition, it is said that G and P satisfy the Markov condition with each other.

If (G, P) satisfies the Markov condition, then P is equal to the product of its conditional distributions of all nodes given values of their parents, whenever these conditional distributions exist. Formally, this is expressed as

$$P(\mathbf{X}) = \prod_{i=1}^{n} P(X_i \mid \mathbf{X}_{pa(i)}). \tag{2.1}$$

2.4 Bayesian networks

Let P be a joint probability distribution of the random variables in some set $\mathbf{X}$, and $G = (\mathbf{X}, \mathbf{E})$ be a DAG. Then, (G, P) is called a *Bayesian network* if (G, P) satisfies the Markov condition. Owing to the properties of the Markov condition, P is the product of its conditional distributions in G, and this is the way P is always represented in a Bayesian network.

A more formal definition is as follows, the term Bayesian network (Pearl (1988)) is used to denote the pair $(m, \boldsymbol{\theta})$ consisting of:

1. The DAG model $m = (\mathbf{X}, \mathbf{E})$ encoding conditional independence statements. X_i is a discrete random variable associated with a node of m.

2. The *parameter* of m denoted by $\boldsymbol{\theta}=(\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_n)$, where $\boldsymbol{\theta}_i$ consists of the local probabilities $\theta_{X_i|\mathbf{X}_{pa(i)}}=P(X_i\mid\mathbf{X}_{pa(i)})$ in Eq. (2.1).

Then Eq. (2.1) can be written as

$$P(\mathbf{X}\mid m,\boldsymbol{\theta})=\prod_{i=1}^{n}P(X_i\mid m,\mathbf{X}_{pa(i)},\boldsymbol{\theta}_i)=\prod_{i=1}^{n}\theta_{X_i|\mathbf{X}_{pa(i)}}. \qquad (2.2)$$

2.5 Learning Bayesian networks

In order to construct a Bayesian network there are two specifications than need to be defined. One is the DAG and the other is the parameters (conditional probability table entries) for that particular DAG. Bayesian networks can be constructed by hand (most common in medical science applications), by data, or a combination of both approaches.

In this thesis, most focus will be on learning Bayesian networks from data. Learning Bayesian networks from data is a difficult task, Cooper (1990) showed that in the worst case it is intractable to compute posterior probabilities in a multiply-connected Bayesian network, this computation being NP-hard. Also, the space complexity of the network increases with its degree of connectivity. Bayesian networks with more edges between their nodes require more storage space for the probability parameters.

In learning from data, most research is concentrated on the learning of optimal DAGs since the parameters can be easily estimated by the empirical conditional frequencies from the data in the case of discrete variables or the use of well-know probability densities in the case of continuous variables.

Given an *i.i.d.* training data set $\mathbf{D} = \{\mathbf{X}_1, \ldots, \mathbf{X}_r, \ldots, \mathbf{X}_N\}$, where $\mathbf{X}_r = \{X_{r,1}, \ldots, X_{r,n}\}$, the goal of structure learning is to find a set of directed edges $\mathbf{E}$ that best models the joint probability distribution $P(X_{r,1}, \ldots, X_{r,n})$ of the data. One common approach is to find the network B that maximises the log-likelihood of the data,

$$LL(B \mid \mathbf{D}) = \sum_{r=1}^{N} \log P_B(\mathbf{X}_r) = \sum_{r=1}^{N} \sum_{i=1}^{n} \log P(X_{r,i} \mid \Pi_{X_{r,i}}). \tag{2.3}$$

Since on average adding an edge never decreases likelihood in the training data, using the log-likelihood as the scoring function can lead to overfitting problems. In order to avoid this, the *Bayesian Scoring Function* (Cooper and Herskovits, 1992; Heckerman *et al.*, 1995) and the MDL principle (Lam and Bacchus, 1994) are commonly used to evaluate structure candidates.

An exhaustive search over all structures can in principle find the best Bayesian network structure, but since the structure space is exponential in the number of nodes in the graph, exhaustive searches are not feasible in more complex networks. Solution methods for intractable cases are usually categorized in independence tests (Neapolitan, 2004) and search based methods (Cooper and Herskovits, 1992; Heckerman *et al.*, 1995). Examples are the K-2 algorithm (Cooper and Herskovits, 1992) that defines a node ordering such that a directed edge can only be added from a high ranking node to a low ranking node. Another is to limit the number of parents a variable can have. This alternative is mostly employed by Bayesian networks for classification tasks since the main objective is the classification performance more than trying to model perfectly the probabilistic dependencies amongst the variables. How to learn simple models efficiently, with a limited number of parents for each variable, for classification, clustering and data visualisation is the main topic of this thesis.

2.6 Recent Bayesian network applications

This section briefly reviews recent applications of Bayesian networks in diverse areas, from Agriculture and Biology through to Medicine and Robotics.

Agriculture

Aitkenhead and Aalders (2007) used evolutionary computation to train a Bayesian network using a dataset containing land cover information and environmental data. Land cover models are useful to understand how the landscape may change in the future, in order to test any land cover change model, existing data must be used but often it is not known which data should be applied to the problem. The dataset used to develop the models included GIS-based data taken from the Land Cover for Scotland 1988 (LCS88), Land Capability for Forestry (LCF), Land Capability for Agriculture (LCA), the soil map of Scotland and additional climatic variables. The proposed evolutionary training approach for the Bayesian network model was capable of deciding automatically what dataset to use in order to obtain optimum (near optimum) results. Two important aspects that the authors were able to conclude were: First is that while there are causal relationships between commonly available datasets and land cover, any approach that is successful in extracting these relationships must be flexible enough to handle multiple data types and sources. Second is that whereas commercial software packages that implement novel modelling methods are useful, they can lack the flexibility required to optimise those modelling methods when the system being modelled contains a large number of variables and uncertain relationships.

Ma and Dai (2005) trained a Bayesian network for predicting the East Asia locust hazard. The training data is obtained by a remote sensing monitor mechanism and experimental conclusions, then by considering the causal relationship among the factors of locust environmental roosts, six label factors indicating the locust disaster are selected to train the Bayesian network. The results show a good agreement between prediction factors and practical factors in the change trend along with time. They conclude that Bayesian networks can provide a new approach for the migratory Locust disaster prediction, as well as for other disaster forecasts. In Granitto *et al.* (2005) the feasibility of using machine vision algorithms for the identification of weed seeds from colour and black and white images was explored. Using standard image-processing techniques 12 features are computed: 6 morphological, 4 colour and 2 textural, their discriminating power as classification features is assessed by a simple Bayesian approach (naive Bayes classifier) and (single and bagged) artificial neural network systems for seed identification. The results indicate that the naive Bayes classifier based on an adequately selected set of classification features has an excellent performance, competitive with that of the comparatively more sophisticated neural network approach.

Aviation

Aircraft accident cases are being modelled with Bayesian networks to identify accident precursors and to assess the potential risk reducing effects of these new products on certain types of accidents. Technology and system concepts to improve aviation safety are presented in Bareither and Luxhøj (2007) by means of introducing methods for Bayesian network analysis with both sensitivity and importance aspects to identify the most influential accident precursors. Luo *et al.* (2005) discussed that it is a key of the formation mechanism analysis to probe into interaction of the factors that cause an aviation calamity in accordance with the characteristics. In order to support the formation mechanism analysis a Bayesian network model is used to study the causality of the factors and the interaction law of human, aircraft, environment and management factors. The relationship between external factors and internal factors were found with formation mechanism analysis of a typical aviation calamity case according to Gauss Bayesian network.

In Greenberg *et al.* (2005) a Bayesian network socio-technical model for investigating the accident rate for multi-crew civil airline aircraft is presented. The model emphasises the influence of airline policy and societal behaviour patterns on the pilots within the piloting system. The main claim that the authors make is that a Bayesian network can be used to bring most aviation safety-critical elements into a common quantitative safety assessment despite the unique problems posed by the very low probability of accidents. They support this claim by replicating certain phenomena such as the low accident rate, the difference between the "more" and "less" safe airlines and other statistical factors of civil aviation. It was found that the model succeeds in explaining the large gap of six to seven orders of magnitude between in-flight measurements of pilots' error and accident rate.

Biology

Accurate and large-scale prediction of protein–protein interactions directly from amino-acid sequences is one of the great challenges in computational biology. In Burger and Nimwegen (2008) a new Bayesian network method is presented that predicts interaction partners using only multiple alignments of amino-acid sequences of interacting protein domains, without tunable parameters, and without the need for any training examples. They demonstrate the power of the proposed method by first applying it to bacterial two-component systems (TCSs) proteins, which are responsible for most signal transduction in bacteria. The results of this experiment provided the first genome-wide reconstruction of two-component signaling networks across all sequenced bacterial genomes. It was found that the proposed model achieved high prediction accuracy when compared with large sets of known interactions. In order to demonstrate the generality of the model, it is applied to a recent data set of about 100 polyketide synthases (PKSs). This application also proved effectively in predicting interaction partners with high accuracy even for relatively small datasets. Also the genome-wide predictions of two-component signaling networks across all sequenced bacteria allowed the authors to make an initial investigation of the structural properties of these networks across bacteria.

BioBayesNet is a new web application, introduced in Nikolajewa *et al.* (2007), that allows the easy modelling and classification of biological data using Bayesian networks. In the learning process of a Bayesian network the user can either upload a set of annotated FASTA format sequences or a set of pre-computed feature vectors. In case of FASTA sequences, the server is able to generate a wide range of sequence and structural features from the sequences. These features are used to learn Bayesian networks. An automatic feature selection procedure assists in selecting discriminative features, providing an (locally) optimal set of features. The output includes several quality measures of the overall network and individual features as well as a graphical representation of the network structure, which allows exploring dependencies between features. Finally, the learned Bayesian network or another uploaded network can be used to classify new data.

McMahon (2005) presented a study on biological invasions. He mentions that one of the key obstacles to better understanding, anticipating, and managing biological invasions is the difficulty in trying to quantify the many important aspects of the communities that affect and are affected by non-indigenous species (NIS). He proposes the use of Bayesian networks to provide an analytical tool for the quantification of communities, since they can determine which components of a natural system influence with others, quantify this influence, and provide inferential analysis of parameter changes when changes in network variables are hypothesized or observed. After a brief explanation of these three functions of BLNs, a simulated network is analysed for structure, parameter estimation, and inference. The proposed approach is explored using a simple forest community example.

Business and Finance

Cowell *et al.* (2007) presented the modeling of operational risk using Bayesian networks. Using an idealised example of a fictitious on line business, they construct a Bayesian network that models various risk factors and their combination into an overall loss distribution. The authors conclude that the main advantages of using

Bayesian networks for modeling operational risk is the facility to incorporate expert opinion through:

- Choice of the variables of interest
- Definition of the structure of the model via the causal dependencies
- Specification of the prior distributions and the conditional probabilities at each node.

Lu *et al.* (2007) presented a study that applies Bayesian network technique to analyse and verify the relationships among cost factors and benefit factors in the development of e-services. Using this technique, the costs involved in moving services online against the benefits received by adopting e-service applications are found. These findings have potential to improve the strategic planning of businesses by determining more effective investment items and adopting more suitable development activities in e-services development. Sun and Shenoy (2007) provided an operational guidance for building naive Bayes Bayesian network models for bankruptcy prediction. First, they suggest a heuristic method that guides the selection of bankruptcy predictors. Based on the correlations and partial correlations among variables, the method aims at eliminating redundant and less relevant variables. Then they develop a naive Bayes model using the proposed heuristic method which is found to perform well based on a 10-fold validation analysis. The developed naive Bayes model consists of eight first-order variables, six of which are continuous. The authors conclude that the results of this study could also be applicable to business decision-making contexts other than bankruptcy prediction.

Mukhopadhyay *et al.* (2006) developed a framework, based on a Bayesian network model, to quantify the risk associated with online business transactions, arising out of a security breach, and thereby help in designing e-insurance products. The output of the Bayesian network is the frequency of occurrence of an e-risk event. Then they come up with a model of the expected loss amount by multiplying the expected loss amount with the probability of occurrence of an event. This model is useful for quantifying e-risk in monetary terms in case of failure of online systems. The authors conclude that insurance companies can use this model to set premium for e-insurance products. Neil *et al.* (2005) described the use of Bayesian networks to model statistical loss distributions in financial operational risk scenarios. Its focus is on modeling "long" tail, or unexpected, loss events using mixtures of appropriate loss frequency and severity distributions where these mixtures are conditioned on causal variables that model the capability or effectiveness of the underlying controls process. They conclude that Bayesian networks can help combine qualitative data from experts and quantitative data from historical loss databases in a principled way. Adopting a Bayesian network-based approach should, therefore, lead to better operational risk governance and a reduced regulatory capital charge. Relying purely on historical loss data and traditional statistical analysis techniques will neither provide good predictions of future operational risk losses, nor a mechanism for controlling and monitoring such losses.

Genetics

Yuan *et al.* (2007) trained naive Bayes classifiers using only the sequence-motif matching scores provided by Beer and Tavazoie's (BT) approach to predict mRNA expression patterns. The simple naive Bayes models were capable of correctly predicting expression patterns for 79% of the genes, based on the same criterion and the same cross-validation procedure as BT, outperforming the 73% accuracy of BT. Chen *et al.* (2007) investigated the combined effects of genetic polymorphisms and non-genetic factors on predicting the risk of Coronary artery disease (CAD) by applying well known classification methods, such as Bayesian networks, naive Bayes, support vector machine, k-nearest neighbours, neural networks and decision trees. The experimental results showed that all these classifiers are comparable in terms of accuracy, while Bayesian networks have the additional advantage of being able to provide insights into the relationships among the variables. The authors observed that the learned Bayesian networks identified many important dependency relationships among genetic variables, which can be verified with domain knowledge. They conclude that conforming to current domain understanding, the results indicate that related diseases (e.g., diabetes and hypertension), age and smoking status are the most important factors for CAD prediction, while the genetic polymorphisms entail more complicated influences.

Needham *et al.* (2006) used Bayesian networks to predict whether or not a missense mutation will affect the function of the protein. The authors support the use of Bayesian networks for this task because they provide a concise representation for inferring models from data, and are known to generalise well to new data. More importantly, they can handle the noisy, incomplete and uncertain nature of biological data. The results showed that the Bayesian network achieved comparable performance with previous machine learning methods. The predictive performance of learned model structures was no better than a naive Bayes classifier. However, analysis of the posterior distribution of model structures allowed biologically meaningful interpretation of relationships between the input variables.

Deng *et al.* (2006) developed a machine learning system for determining gene functions from heterogeneous data sources using a Weighted Naive Bayesian network (WNB). The knowledge of gene functions is crucial for understanding many fundamental biological mechanisms such as regulatory pathways, cell cycles and diseases. The authors comment that a major goal for them is to accurately infer functions of putative genes or Open Reading Frames (ORFs) from existing databases using computational methods. However, this task is difficult since the underlying biological processes represent complex interactions of multiple entities. Therefore, many functional links would be missing when only one or two sources of data are used in the prediction. Their hypothesis is that integrating evidence from multiple and complementary sources could significantly improve the prediction accuracy. The experimental results suggest that the above hypothesis is valid, as well as guidelines for using the WNB system for data collection, training and predictions. The combined training data sets contain information from gene annotations, gene expressions, clustering outputs, keyword annotations, and sequence homology from public databases. The current system was trained and tested on the genes of budding yeast *Saccharomyces cerevisiae*. The WNB model can also be used to analyse the contribution of each source of information toward the prediction performance through the weight training process. The authors conclude that contribution analysis using the WNB could potentially lead to significant scientific discovery by facilitating the interpretation and understanding of the complex relationships between biological entities.

Rodin and Boerwinkle (2005) introduced the method of Bayesian networks to the domain of genotype-to-phenotype analyses and provide an example application. The proposed Bayesian network method was applied to 20 single nucleotide polymorphisms (SNPs) in the apolipoprotein (apo) E gene and plasma apoE levels in a sample of 702 individuals from Jackson, MS. The plasma apoE level was the primary target variable. After some analysis it was found that the edge between SNP 4075, coding for the well-known ε2 allele, and plasma apoE level was strong.

Pudimat *et al.* (2005) developed a probabilistic modeling approach, which allows considering diverse characteristic binding site properties to obtain more accurate representations of binding sites. Commonly used stochastic models for binding sites are position-specific score matrices (PSSM), which show weak predictive power. In the proposed work the binding site properties are modelled as random variables in Bayesian networks, which are capable of dealing with dependencies among the variables. In order to compare the predictive power of the proposed method to that of PSSM models, cross-validation tests on two datasets was performed namely MEF-2 binding sites, AP-1 boxes and Sp1 binding sites. In all cases, considerable improvements were found in the classification error rates using the Bayesian network models.

Manufacturing

Cheah *et al.* (2007) proposed a Bayesian network for the representation and reasoning of the manufacturing-environmental knowledge for assembly design decision support. They also propose a methodology for the indirect knowledge acquisition, in order to build the network, using fuzzy cognitive maps and for the conversion of the representation into a Bayesian network. In Chen and Chang (2007) a Bayesian network is used to implement a data mining task for computer integrated manufacturing (CIM). The Bayesian network was capable of finding the cause factors in various parameters that affect in the semiconductor cleaning process. In Li and Shi (2007) a causal modelling approach using Bayesian networks is proposed to improve an existing causal discovery algorithm by integrating manufacturing domain knowledge with the algorithm. The proposed model is demonstrated by discovering the causal relationships among the product quality and process variables in a rolling process. The authors conclude that the results of the casual model, when allied with engineering interpretations, can be used to facilitate rolling process control.

Weber and Jouffe (2006) presented a methodology for developing Dynamic Object Oriented Bayesian Networks (DOOBNs) to formalise complex manufacturing processes in order to optimise the diagnosis and the maintenance policies. The added value of the proposed formalisation methodology consists in using the a priori knowledge of both the system's functioning and malfunctioning. The Bayesian networks are built on principles of adaptability and integrate uncertainties on the relationships between causes and effects. The methodology is tested, in an industrial context, to model the reliability of a water (immersion) heater system. They find that the obtained model is more compact and readable than the Markov Chain model. Furthermore, the dependency between several failure modes of a component and common modes is easily modelled by the Bayesian network. The authors conclude that DOOBNs represent a very powerful tool for decision-making in maintenance.

Perzyk *et al.* (2005) studied the modeling capabilities of two types of learning systems: the naive Bayesian classifier (NBC) and artificial neural networks (ANNs). Comparisons are done, using simulated and real industrial data, based on their prediction errors and relative importance factors of input signals. It was found that NBC can be an effective and, in some applications, a better tool than ANNs.

Medicine

Verduijn *et al.* (2007) introduced a prognostic model that builds on the Bayesian network methodology. Prognostic models, given a set of patient specific parameters, predict the future occurrence of a medical event or outcome. Example events are the occurrence of specific diseases (e.g., cardiovascular diseases and cancer) and death. The authors identified three main limitations with previous prognostic models which use supervised data analysis methods, these are: First, attribute selection before inducing a model, often removing many attributes that are deemed relevant for prognosis. Second, the resulting models regard prognosis to be a one-time activity at a predefined time. And third, the models impose fixed roles of predictor (independent variable, input) and outcome variable (dependent variable, output) to the attributes involved. To overcome these limitations, they propose a prognosis model based on Bayesian network methodology (PBN), which provides a structured representation of a health care process by modelling the mutual relationships among variables that come into play in the subsequent stages of the care process and the outcome. As a result, the PBN allows for making predictions at various times during a health care process. Also, prognostic statements are not limited to outcome variables, but can be obtained for all variables that occur beyond the time of prediction.

Suebnukarn and Haddawy (2006) developed representational techniques and algorithms for generating tutoring hints in problem-based learning (PBL) using Bayesian networks, implemented in a collaborative intelligent tutoring system called COMET. The system uses Bayesian networks to model individual student clinical reasoning, as well as that of the group. In order to evaluate the performance of the system, the tutoring hints generated by COMET with those of experienced human tutors were compared. The results showed that on average, 74.17% of the human tutors used the same hint as COMET. The authors conclude that Bayesian network clinical reasoning models can be combined with generic tutoring strategies to successfully emulate human tutor hints in group medical PBL.

Alvarez *et al.* (2006) performed a study to evaluate the feasibility of using a Bayesian network to improve the accuracy of diagnosing pyloric stenosis. Records of 118 infants undergoing an ultrasound to rule out pyloric stenosis were reviewed. Data from 88 (75%) infants were used to train a Bayesian decision network that predicted the probability of pyloric stenosis. The emergency department records of the remaining 28 (25%) infants were used to test the network. The results showed that physicians using the Bayesian decision network predicted better the probability of pyloric stenosis among infants in the testing set than those not using the network. Physicians using the network would have ordered 22% fewer ultrasounds and missed no cases of pyloric stenosis. The authors concluded after their study that the use of a Bayesian decision network may improve the accuracy of physicians diagnosing infants with possible pyloric stenosis. The use of this decision tool may safely reduce the need for imaging among infants with suspected pyloric stenosis.

Visscher *et al.* (2005) used a Bayesian network as a primary tool for building a decision-support system, diagnosis and treatment, for the clinical management of ventilator-associated pneumonia. While in Mani *et al.* (2005) the practical aspects of building a Bayesian Network from a large medical dataset for Mental Retardation is discussed.

In traditional Chinese medicine (TCM), Bayesian networks have had several applications recently as well. Zou *et al.* (2007) combined Principal Component Analysis with a Bayesian network classifier in order to perform syndrome classification of chronic gastritis. Wang and Wang (2006) proposed a quantitative method based on Bayesian networks to predict syndrome automatically. The method uses Bayesian networks instead of using rules, which differentiates from other existing quantitative methods in TCM. The results show that the diagnostic model obtains relative reliable predictions of syndrome, and its average predictive accuracy rate reaches 91.68%, which makes the proposed method feasible, effective and useful in the modernization of TCM. In Zhu *et al.* (2006) a Bayesian network was used to study the relationship between the key elements of syndrome and the symptoms, and the relationship among different key elements, in which the computing diagnosis result were identical to the result from an experienced TCM doctor. The study showed that the Bayesian network is capable of dealing effectively with the information of symptoms and signs for syndrome differentiation, although it did not reflect comprehensively the thinking ability of TCM doctors in doing syndrome differentiation. In Wang and Zong (2006) to classify tongue types automatically (tongue recognition is one of the most important components in TCM), they proposed a quantitative method based on Bayesian networks to build the mapping relationships between tongue images and tongue types. This system includes novel features that help to overcome main disadvantages of routine TCM expert system obtaining encouraging results that can yield this system to be used as an assistant diagnostic tool.

Robotics

Lazkano *et al.* (2007) presented the use of Bayesian networks as a learning technique to manage task execution in mobile robotics. To learn the Bayesian Network structure from data, the K2 structural learning algorithm is used, combined with three different net evaluation metrics. The experiment led to a new hybrid multi-classifying system resulting from the combination of 1-NN with the Bayesian Network. As an application example of the proposed method, a door-crossing behaviour in a mobile robot using only sonar readings, in an environment with smooth walls and doors is presented. Both the performance of the learning mechanism and the experiments run in the real robot-environment system show that Bayesian Networks are valuable learning mechanisms, able to deal with the uncertainty and variability inherent to such systems.

Infantes *et al.* (2006) proposed the use of dynamic Bayesian networks as models of robotic tasks behaviours. In many situations, these tasks involve complex behaviours combining different functionalities (e.g. perception, localisation, motion planning, and motion execution). The dynamic Bayesian network formalism allows learning and controlling behaviours with controllable parameters. The proposed approach is experimented on a real robot, where the model of a complex navigation task is learned over a large number of runs using a modified version of Expectation Maximisation for dynamic Bayesian network. The resulting dynamic Bayesian network is then used to control the robot navigation behaviour. It is shown that for some given objectives (e.g. avoid failure), the learned dynamic Bayesian network driven controller performs much better (one order of magnitude less failure) than the programmed controller. The authors conclude that the proposed approach remains generic and can be used to learn complex behaviours other than navigation and for other autonomous systems.

Scene understanding is an important problem in intelligent robotics. In Im and Cho (2006) a context-based Bayesian network method with Scale-Invariant Feature Transform (SIFT) for scene understanding is proposed. The use of a Bayesian network for this problem is chosen because of its robustness to manage the uncertainty, and powerfulness to model high-level contexts like the relationship between places and objects. The proposed approach consists first in an image pre-processing step which extracts features from vision information and the objects existence information is extracted by SIFT that is rotation and scale invariant. This information is provided to Bayesian networks for robust inference in scene understanding. Experiments carried out in real university environment showed that the Bayesian network approach using visual context-based low level feature and high level object context, which is extracted by SIFT, is effective.

Wang and Ramos (2005) used the Bayesian Structural EM algorithm as a classification method to learn and interpret hyperspectral sensor data in robotic planetary missions. Many spacecraft carry spectroscopic equipment as wavelengths outside the visible light in the electromagnetic spectrum give much greater information about an object. The algorithm presented combines the standard Expectation Maximisation (EM), which optimises parameters, with structure search for model selection. The Bayesian Information Criterion (BIC) score is used to learn the network structure. The EM procedure only assures convergence to a local maxima, thus, it is required a good initial graph structure. Two initial structures are used: the naive Bayes, and the tree-augmented-naive Bayes (TAN) structures. Preliminary experiments showed that the former results in a structure that can only determine the presence and types of minerals with merely 13% accuracy while the latter results in a structure that has approximately 94% accuracy.

2.7 Summary

This chapter has reviewed the theory and practical applications of Bayesian networks to provide general background information for the research reported in subsequent chapters of the thesis.

Chapter 3

SIMPLE BAYESIAN NETWORK CLASSIFIERS

3.1 Preliminaries

Classification is the task of identifying the class label of instances which are described by a set of attributes (features). The construction of classifiers from data is an active research topic in the machine learning and data mining community. Popular techniques for learning classifiers from data include: decision-trees, artificial neural networks, support vector machines, rule induction and fuzzy systems. Although Bayesian networks are probabilistic graphical models useful for performing inference under conditions of uncertainty and knowledge representation (Pearl, 1988), they were not considered as classifiers until a very simple Bayesian network called the *naive Bayesian classifier* (Duda and Hart, 1973) proved to be effective (Langley *et al.*, 1992). The naive Bayesian classifier makes a strong independence assumption amongst its attributes, assuming that all attributes are conditionally independent given the classification node. In order to overcome this independence assumption, the augmenting of the naive Bayesian model with edges between attributes has yielded many types of Bayesian network classifiers which will be presented later on. The most cited and studied augmenting technique for the naive Bayesian classifier so far is

Friedman *et al.*'s *Tree Augmented Naive Bayes* (TAN) classifier (Friedman *et al.*, 1997). The TAN algorithm uses a variant of the Chow and Liu (Chow and Liu, 1968) method for learning tree-like structures in which the class variable has no parents and each attribute has one other parent in addition to the class variable. This restriction is very convenient since it makes it an algorithm which is computationally efficient while still optimising likelihood. Apart from the computational advantages, networks with low connectivity also possess conceptual advantages in the sense that by having simpler topologies, the underlying causal and probabilistic relationships in the domain are easier to understand.

On the other hand, the use of ***multiply-connected networks*** allows a much better approximation of the underlying distribution, but with several disadvantages. Cooper (1990) showed that in the worst case it is intractable to compute posterior probabilities in a multiply-connected Bayesian network, this computation being NP-hard. Also, the space complexity of the network increases with its degree of connectivity. Bayesian networks with more connections (edges) between their nodes require more storage space for the probability parameters. The number of probability parameters needed for each node increases exponentially with the number of its parents (incoming edges). The trade-off between accuracy (more complex models) and usefulness (simpler models) was explored by Lam and Bacchus (1994). They presented an approach for learning unrestricted multiply-connected Bayesian networks based on Rissanen's *Minimum Description Length* (MDL) principle (Rissanen, 1978), which says that the best model of a collection of data is the one that minimises the sum of the encoding lengths of the data and the model itself. However, finding the network that minimises the sum of these two components is computationally intractable (Chickering, 1995), so they use a heuristic search algorithm that tries to find a network that has low, but not necessarily minimum, description length. The experiments showed good results, but the MDL approach was studied in Friedman *et al.* (1997) for learning Bayesian network classifiers yielding poor results, especially for data sets that had more than fifteen attributes. Their analysis suggested that a network with a high MDL score is not necessarily a better classifier.

This chapter presents a new learning method for constructing Bayesian network classifiers with a Bayesian approach to handle the trade-off between accuracy and complexity that can be implemented incrementally. This type of network has the same restrictions as the TAN model, in that the class variable has no parents and each attribute has, at the most, one other parent in addition to the class variable. However, it differs from the TAN model since the resulting network does not have to be a spanning tree. In the proposed approach, the number of edges is automatically regulated depending on the number of training patterns, yielding much more reliable networks according to the training data. The resulting structures can vary from the simple naive Bayesian classifiers, with zero augmenting edges between attributes, to the most complex one, with the restriction mentioned above, which is the TAN classifier with $n-1$ augmenting edges, where n is the number of attributes. Although the resulting network can have $e \leq n-1$ edges, the network is still optimum in the sense of divergence (cross-entropy), likelihood, and the Bayesian probability of error for the e augmenting edges class network. Experiments on benchmark data sets to show how the size of the training set influences the resulting structures as well as the classification performance compared to the naive Bayesian and TAN models, have been carried out. Also the proposed method was compared to state of the art classifiers such as the C4.5 and a MLP neural network in a real-world industrial application.

This chapter is organised as follows: Section 3.2 presents a description of Bayesian network classifiers. In Section 3.3, the proposed learning method is presented, starting from the Bayesian approach to handle the trade-off between accuracy and complexity up to presenting the construction method of the networks. Section 3.4, describes the methods used for carrying out the experiments to test the new simple Bayesian network classifiers. Results are shown in Section 3.5 and the summary of this work in Section 3.6.

3.2 Bayesian network classifiers

The goal of a Bayesian network classifier is to correctly predict the class label C given a vector of attributes $\{X_1,\ldots,X_n\}$. If the performance is measured on the correct classification percentage on a test set, then the optimal prediction for $\{X_1,\ldots,X_n\}$ is the class that maximises $P(C\,|\,X_1,\ldots,X_n)$ (Duda and Hart, 1973). A Bayesian network classifier models the joint probability distribution $P(X_1,\ldots,X_n,C)$ and converts it to conditional distribution using the Bayes' theorem. The simplest of the Bayesian network classifiers is the *naive Bayesian classifier* where the class variable has no parents and each attribute has only the class as its parent (see Fig. 3.1). This strong independence assumption is overcome by incorporating edges between attribute nodes. Firedman *et al.* (1997) TAN algorithm is one of the most popular techniques for augmenting the naive Bayes model. The TAN algorithm uses a variant of Chow and Lui's seminal work for finding optimal tree-like structures where each variable has one other parent in addition to the class (see Fig. 3.2). In what follows, other research on Bayesian network classification is presented.

Cerquides and Lopez de Mantaras (2005) presented several Bayesian algorithms for learning TAN models. They introduced decomposable distributions over TANs, extending the work of Meila and Jaakkola (2000) to TANs, allowing the computation of the exact Bayesian model averaging over TAN structures and parameters in polynomial time. These classifiers provide significant improvements, especially when data is scarce. Acid *et al.* (2005) developed a new "score+search"-based algorithm for learning Bayesian network classifiers from a database, showing that score-based learning specialised for classification can compete favourably with state-of-the-art Bayesian network classifiers. An empirical comparison of four Bayesian networks classifiers was carried out in Cheng and Greiner (1999). The results showed that conditional independence-based learning algorithms were competitive with the best known classifiers based on both Bayesian networks and other formalisms. Shi and Huang (2002) proposed a new algorithm for constructing TAN models which is linear in the number of attributes. This is favourable when working with high dimensional data since the original TAN algorithm is quadratic in the number of attributes. In Monti and Cooper (1999) a new Bayesian network model for classification that combines the naive Bayes classifier and the finite mixture classifier is presented. The classifier is obtained by superimposing a finite mixture model on the set of attributes of a naive Bayes model. Experimental results showed that the new classifier can often outperform the naive Bayes model in terms of classification accuracy, while significantly improving the calibration of the probability estimates. Grossman and Domingos (2004) showed that choosing structures by maximising conditional likelihood, while setting parameters by maximum likelihood, yields good results, with better probability estimates than the naive Bayes, TAN, and generatively-trained Bayesian networks. Keogh and Pazzani (2002) developed an algorithm for constructing TAN-type classifiers using classification accuracy rather than maximum likelihood scores. In Hwang and Zhang (2005) a Bayesian model averaging network classifiers over several distinct node orders, obtained using the Markov Chain Monte Carlo sampling technique, was proposed, proving to be especially effective when the given dataset was very sparse. Jing *et al.* (2005) showed that an effective Bayesian network classifier can be constructed by parameter boosting coupled with discriminative structuring learning, outperforming the naive Bayes, TAN, and other Bayesian network classifiers on benchmark data sets. In Kleiner and Sharp (2000) a new architecture for the induction of classifiers based on Bayesian networks is presented. Experiments showed that the new algorithm outperformed the TAN model in the case of data with weak or multiple correlations.

The following section presents a new approach to augment the naive Bayes classifier using a Bayesian approach to handle the trade-off between accuracy and complexity.

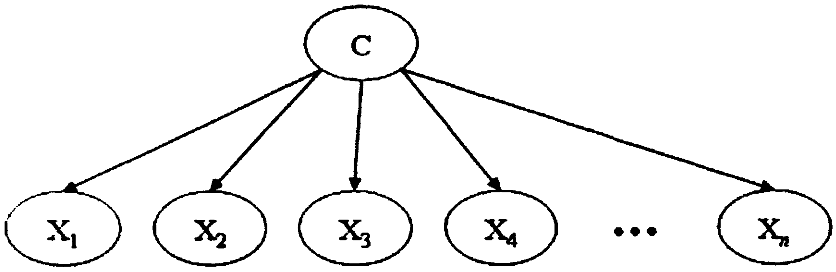

Figure 3.1 Naive Bayes classifier

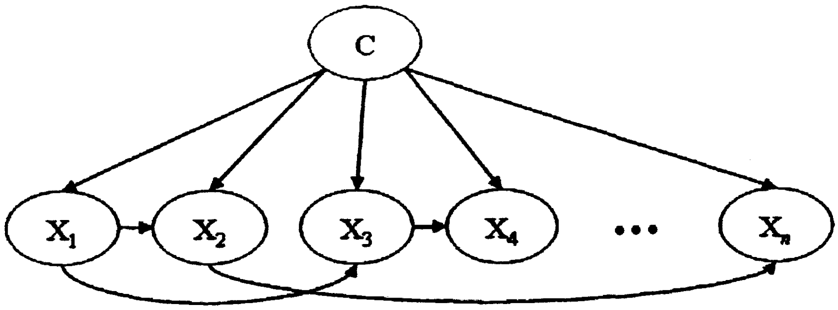

Figure 3.2 TAN classifier

3.3 Training simple Bayesian network classifiers

The original TAN classifier is constructed by the *maximum weighted spanning tree* MWST using Kruskal's algorithm (Kruskal, 1956), for example, with the conditional mutual information as weights. This approach is optimum since it maximises likelihood. Nevertheless, for n attributes nodes, the TAN will augment the naive Bayes classifier with $n-1$ edges regardless if the amount of training data available is enough to support that number of edges, thereby ignoring the balance between complexity and accuracy. This section presents a simple Bayesian network classifier which can be implemented incrementally, introducing a Bayesian complexity measure that regulates automatically the amount of augmented edges permitted to augment the naive Bayes classifier depending on the amount of data available. The resulting classifier will end up with a structure that lies in between the naive Bayes and the TAN, including both of these models as well.

First a simple encoding method for the network is presented, then a complexity measure is introduced using Bayes' Theorem. Finally, the incremental learning of the simple Bayesian network classifier is described.

3.3.1 Network encoding

For a Bayesian network with n attribute nodes, let the representation of the network be constructed using $n+1$ symbols (one for each attribute node plus an additional symbol to represent the stop command) of length $m = \log_2(n+1)$ bits.

Let a Bayesian network B encoding be represented by a sequence of ordered pairs:

$$(n_1, n_2)(n_3, n_4)(n_5, n_6)\ldots(n_{s-1}, n_s)Stop,$$

where an ordered pair (n_i, n_j) is a code $C_{ij} = C(n_i)C(n_j)$ formed by the concatenation of the code representing the node n_i and the code representing n_j. Each ordered pair in the sequence represents the nodes that have an edge between them. For example, the Bayesian network of Fig. 3.3 that has 3 nodes. Each node can be represented by codes of length $m = \log_2(3+1) = 2$ bits. Then, let $C(X_1) = 00$, $C(X_2) = 01$, $C(X_3) = 10$, and $Stop = 11$ be the codes of each node and the stop command respectively.

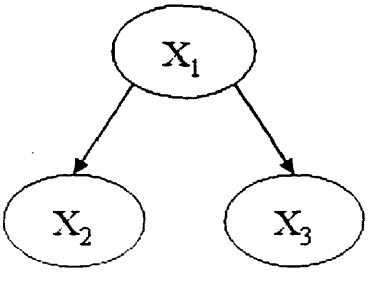

Figure 3.3 An example of a Bayesian network with 3 nodes and 2 edges.

The encoding of the Bayesian network of Fig. 3.3 is therefore $(X_1, X_2)(X_1, X_3)Stop = 0001001011$, with codelength $l(B) = 10$ bits.

It is important to notice that this is not the only possible encoding scheme. It is, nevertheless, simple and it performs well in the experimental tests.

3.3.2 Complexity measure

In order for a simple model to be replaced by a more complex one, the amount of training data is fundamental. This is because a more complex model requires a greater amount of data for its validation. All of this is taken into account correctly in the Bayes' formula:

$$P(B \mid \mathbf{D}) = \frac{P(\mathbf{D} \mid B)P(B)}{P(\mathbf{D})}, \tag{3.1}$$

where the likelihood $P(\mathbf{D} \mid B)$ will usually increase with more complex models. On the other hand, the prior $P(B)$ will decrease with the complexity since complex models are less probable. Nevertheless, no matter how complex the model is, if there is enough training data to support it, the sum of the likelihood for each instance of the training set will be high enough to compensate for the very low prior probability.

So, for a simple model B_s to be replaced by a more complex one B_c, the following criteria must hold:

$$\frac{P(\mathbf{D} \mid B_s)P(B_s)}{P(\mathbf{D} \mid B_c)P(B_c)} < 1. \tag{3.2}$$

The ratio $P(\mathbf{D} \mid B_s) / P(\mathbf{D} \mid B_c)$ is known as the *Bayes* Factor (Kass and Raftery, 1995). Applying the base 2 logarithm,

$$k = \log_2 P(\mathbf{D} \mid B_s) - \log_2 P(\mathbf{D} \mid B_c) + \log_2 P(B_s) - \log_2 P(B_c) \tag{3.3}$$

with the condition being $k < 0$ to replace B_s by B_c.

The prior of the Bayesian network can be expressed using the network's encoding system. It is known from coding theory (Mackay, 2003) that probabilities can be mapped to optimal codelenghts, a uniquely decodeable code that minimises the expected codelength. The expected length is minimised only if the codelengths $l(M_i)$ for a given model M_i are equal to the *Shannon information* contents: $l(M_i) = \log_2(1/P(M_i))$. So, for a Bayesian network B with description length $l(B)$, the prior can be estimated by

$$P(B) = 2^{-l(B)}, \tag{3.4}$$

and if the data $\mathbf{D} = \{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$ is *i.i.d.*, then equation (3.3) can be written as

$$k = l(B_c) - l(B_s) + \sum_{r=1}^{N} \sum_{i=1}^{n} \log_2 P(X_{r,i} \mid B_s) - \log_2 P(X_{r,i} \mid B_c). \tag{3.5}$$

Then, let the structure of B_c be the same as B_s but with one additional edge. This means that the description length of B_c uses two more symbols than B_s, i.e., $l(B_c) - l(B_s) = 2m$. Thus, if the extra edge in B_c is due to attribute X_ω being parent of attribute X_υ and if the decomposability of a Bayesian network given by (2.2) is considered, equation (3.5) can be expressed as

$$k = 2m + \sum_{r=1}^{N} \log_2 P(X_{r,\upsilon} \mid C_r) - \log_2 P(X_{r,\upsilon} \mid X_{r,\omega}, C_r). \tag{3.6}$$

Equation (3.6) presents a way to measure the effect of adding an extra edge to a naive Bayesian network classifier. Negative values of k indicate that there is enough data to support that extra edge. The next step is to find a stopping condition for the augmenting of the naive Bayes classifier. For this, comparing the effect of having no edges between attributes (naive Bayes) against having e augmenting edges is required. Because of the logarithmic property (multiplicative terms become additive ones) the equation to compute, K_e, that indicates if there is enough data to support e edges compared to 0 edges (naive Bayes), is

$$K_e = \sum_{i=1}^{e} k_i, \tag{3.7}$$

where k_i represents the k value for the i^{th} edge being considered. Then, the adding of edges will continue while $K_e < 0$. When K_e becomes positive, the network does not have sufficient data to support the adding of any more edges. The fact that K_e can be computed by the sum of single added edges makes this method computationally fast. It also allows the computation to be incremental, as for each iteration the value of K_e can be tested.

3.3.3 Simple learning algorithm

Since the learning of the Chow-Liu trees present an efficient way of maximising the likelihood with simple tree structures (each attribute has one more parent in addition to the class), an adaptation of this method can be used to augment incrementally the naive Bayesian classifiers. This yields networks where the number of augmenting edges will depend on the amount of training data, not restricted to $n-1$ edges as is the TAN model.

In order to assess the accuracy of the approximation of the joint probability distribution made by the Bayesian network, the Kullback-Leibler divergence or *cross-*

entropy is used. Let $P(\mathbf{X})$ and $P_B(\mathbf{X})$ be two probability distributions of n discrete variables $\mathbf{X} = \{X_1, \ldots X_n\}$. The cross-entropy is defined by

$$KL(P(\mathbf{X}) \,||\, P_B(\mathbf{X})) = \sum_{\mathbf{X}} P(\mathbf{X}) \log \frac{P(\mathbf{X})}{P_B(\mathbf{X})}, \tag{3.8}$$

and has the property that $KL(P(\mathbf{X}) \,||\, P_B(\mathbf{X}))$ is non-negative, and equal to 0 if and only if $P(\mathbf{X}) = P_B(\mathbf{X})$ for all $\mathbf{X}$.

For the learning procedure, let $X_1, \ldots, X_n$ be a set of attributes variables and C be the class variable. Also, let B be an augmented naive Bayes model with $e \leq n-1$ edges, so $\Pi_C = \emptyset$ and there is a function π that defines a network over $X_1, \ldots, X_n$ such that $\Pi_{X_i} = \{C, X_{\pi(i)}\}$ if $\pi(i) > 0$, and $\Pi_{X_i} = \{C\}$ if $\pi(i) = 0$. This learning procedure can be formulated as an optimisation problem consisting in finding a network defining function π over $X_1, \ldots, X_n$ such that the cross-entropy between the original joint probability distribution of the data and the approximation made by the Bayesian network is minimised.

It is shown later that the procedure *Build Network* solves this optimisation problem, finding the optimal network with $e \leq n-1$ augmenting edges amongst all the possible networks with that number of edges. This procedure follows the general outline of the TAN procedure called *Construct-TAN*, except that now the number of augmenting edges is automatically regulated by (3.7).

The conditional mutual information between attributes given the class variable is employed by the *Build Network* procedure. This function is defined as

$$I(\mathbf{X};\mathbf{Y}\mid\mathbf{Z}) = \sum_{\mathbf{x},\mathbf{y},\mathbf{z}} P(\mathbf{x},\mathbf{y},\mathbf{z}) \log \frac{P(\mathbf{x},\mathbf{y}\mid\mathbf{z})}{P(\mathbf{x}\mid\mathbf{z})P(\mathbf{y}\mid\mathbf{z})}. \tag{3.9}$$

This function measures the information that **Y** provides about **X** when the value of **Z** is known, and $I(\mathbf{X};\mathbf{Y}\mid\mathbf{Z})$ is non-negative.

The *Build Network* procedure consists of the following steps:

1. Compute $I(X_i;X_j \mid C)$ between each pair of attribute variables, $i \neq j$.
2. Construct a complete undirected graph in which the nodes are the attribute variables $X_1,\ldots,X_n$. Assign the weight of an edge connecting X_i to X_j by $I(X_i;X_j \mid C)$.
3. Apply the MWST algorithm, for each iteration (adding of an edge) $K_e < 0$ must hold. If K_e becomes positive, the MWST algorithm is stopped.
4. Transform the resulting undirected network to a directed one by choosing a root variable and then setting the direction of all edges to be outward from it.
5. Construct an augmented naive Bayes model by adding a vertex labeled C and adding an edge from C to each X_i.

Notice that in step 3, if K_e is positive for the first edge selected by the MWST algorithm, then the resulting structure will be that of the naive Bayes classifier. On the other hand, if $K_e < 0$ throughout the entire algorithm, then the resulting network will be the TAN model, having $n-1$ edges.

Theorem I: Let **D** be a collection of N instances of $X_1,\ldots,X_n,C$ which are *i.i.d.* Amongst all the network structures with $e \leq n-1$ edges, the procedure *Build Network* generates an augmented naive Bayes classifier that minimises the cross-entropy (Kullback-Leibler divergence) between the real joint probability distribution P, from where $\mathbf{D}$ was acquired, and the approximation of the joint probability distribution P_B made by the Bayesian network.

Proof: Let $\mathbf{X} = \{X_1, \dots X_n, C\}$, then using the definition of the cross-entropy (3.8),

$$KL(P(\mathbf{X}) \,||\, P_B(\mathbf{X})) = \sum_{\mathbf{X}} P(\mathbf{X}) \log P(\mathbf{X}) - \sum_{\mathbf{X}} P(\mathbf{X}) \log P_B(\mathbf{X}),$$

since the joint probability distribution of the Bayesian network can be computed using (2.2), then

$$= -\sum_{\mathbf{X}} P(\mathbf{X}) \sum_{i}^{n} \log P(X_i \,|\, \Pi_{X_i}) - H(\mathbf{X}), \tag{3.10}$$

with $H(\mathbf{X}) = -\sum_{\mathbf{X}} P(\mathbf{X}) \log P(\mathbf{X})$.

Equation (3.10) can be adapted for the augmented naive Bayes with $e \leq n-1$ edges model as follows. Let B be a model where the parents of X_i are defined by π, so $P(X_i \,|\, \Pi_{X_i}) = P(X_i \,|\, X_{\pi(i)}, C)$ if $\pi(i) > 0$ and $P(X_i \,|\, \Pi_{X_i}) = P(X_i \,|\, C)$ if $\pi(i) = 0$. Therefore, (3.10) can be expressed as

$$= -\sum_{\mathbf{X}} P(\mathbf{X}) \sum_{i,\pi(i)>0} \log P(X_i \,|\, X_{\pi(i)}, C) - \sum_{\mathbf{X}} P(\mathbf{X}) \sum_{i,\pi(i)=0} \log P(X_i \,|\, C) - H(\mathbf{X})$$

$$= -\sum_{\mathbf{X}} P(\mathbf{X}) \sum_{i,\pi(i)>0} \log \frac{P(X_i, X_{\pi(i)} \,|\, C)}{P(X_i \,|\, C) P(X_{\pi(i)} \,|\, C)} - \sum_{\mathbf{X}} P(\mathbf{X}) \sum_{i,\pi(i)=0} \log \frac{P(X_i, C)}{P(X_i) P(C)} \tag{3.11}$$

$$-\sum_{\mathbf{X}} P(\mathbf{X}) \sum_{i,\pi(i)>0} \log P(X_i \,|\, C) - \sum_{\mathbf{X}} P(\mathbf{X}) \sum_{i,\pi(i)=0} \log P(X_i) - H(\mathbf{X}).$$

$P(X_i)$, $P(X_i,C)$, and $P(X_i,X_{\pi(i)},C)$ can be computed as marginal probabilities from $P(\mathbf{X})$, so then

$$-\sum_{\mathbf{X}} P(\mathbf{X})\log P(X_i) = -\sum_{X_i} P(X_i)\log P(X_i) = H(X_i),$$

$$-\sum_{\mathbf{X}} P(\mathbf{X})\log P(X_i \mid C) = -\sum_{X_i,C} P(X_i,C)\log P(X_i \mid C) = H(X_i \mid C),$$

and

$$\sum_{\mathbf{X}} P(\mathbf{X})\log \frac{P(X_i,X_{\pi(i)} \mid C)}{P(X_i \mid C)P(X_{\pi(i)} \mid C)}$$

$$= \sum_{X_i,X_{\pi(i)},C} P(X_i,X_{\pi(i)},C)\log \frac{P(X_i,X_{\pi(i)} \mid C)}{P(X_i \mid C)P(X_{\pi(i)} \mid C)} = I(X_i;X_{\pi(i)} \mid C),$$

$$\sum_{\mathbf{X}} P(\mathbf{X})\log \frac{P(X_i,C)}{P(X_i)P(C)} = \sum_{X_i,C} P(X_i,C)\log \frac{P(X_i,C)}{P(X_i)P(C)} = I(X_i;C).$$

Thus, (3.11) becomes

$$KL = -\sum_{i,\pi(i)>0} I(X_i;X_{\pi(i)} \mid C) - \sum_{i,\pi(i)=0} I(X_i;C) + \sum_{i,\pi(i)>0} H(X_i \mid C) + \sum_{i,\pi(i)=0} H(X_i) - H(\mathbf{X}), \tag{3.12}$$

and by using the following property (Cover and Thomas, 1991): $I(\mathbf{X};\mathbf{Y}) = H(\mathbf{X}) - H(\mathbf{X} \mid \mathbf{Y})$, the KL is

$$KL = -\sum_{i,\pi(i)>0} I(X_i;X_{\pi(i)} \mid C) - \sum_i I(X_i;C) + \sum_i H(X_i) - H(\mathbf{X}). \tag{3.13}$$

Now, because $I(X_i;C)$, $H(X_i)$, and $H(\mathbf{X})$ for all i are independent of the resulting structure and $I(X_i;X_{\pi(i)} \mid C)$ is non-negative, minimising the cross-entropy is equivalent to maximising the term

$$\sum_{i,\pi(i)>0} I(X_i;X_{\pi(i)}\mid C).$$

Since the branch weights are additive, the maximum-weight dependence network with $e \leq n-1$ edges can then be constructed branch by branch. This can be achieved directly by using Kruskal's MWST algorithm which is greedy and that the *Build Network* procedure employs.

□

The *Build Network* procedure also maximises the log likelihood between the Bayesian network and the data.

Theorem II: Let **D** be a collection of N instances of $X_1,\ldots,X_n,C$ which are *i.i.d.* Amongst all the network structures with $e \leq n-1$ edges, the procedure *Build Network* generates an augmented naive Bayes classifier that maximises the log likelihood $LL(B\mid \mathbf{D})$.

Proof: In Friedman *et al.* (1997) they derive that:

$$LL(B\mid \mathbf{D}) = N\sum_{X_i} I(X_i;\Pi_{X_i}) + \text{ constant term.} \tag{3.14}$$

So, maximising the log likelihood is equivalent to maximising the term

$$\sum_{X_i} I(X_i;\Pi_{X_i}).$$

This term can be adapted for the augmented naive Bayes with $e \leq n-1$ edges model as follows. Let B be a model defined by $\pi(\cdot)$. Since the class node C is the vertex of the structure (no parents), then $I(C;\Pi_C)=0$. As stated previously, the parents of X_i

are defined by π , so $I(X_i;\Pi_{X_i})=I(X_i;X_{\pi(i)},C)$ if $\pi(i)>0$ and $I(X_i;\Pi_{X_i})=I(X_i;C)$ if $\pi(i)=0$ .Therefore, the term to be maximised is,

$$\sum_{i,\pi(i)>0} I(X_i;X_{\pi(i)},C)+\sum_{i,\pi(i)=0} I(X_i;C). \tag{3.15}$$

Then, by using the chain law of mutual information (Cover and Thomas, 1991): $I(\mathbf{X};\mathbf{Y},\mathbf{Z})=I(\mathbf{X};\mathbf{Z})+I(\mathbf{X};\mathbf{Y}\mid\mathbf{Z})$, (3.15) can be rewritten as

$$\sum_{i} I(X_i;C)+\sum_{i,\pi(i)>0} I(X_i;X_{\pi(i)}\mid C).$$

Notice that the first term is not affected by the structure of the network. Thus, only the second term needs to be maximised. The *Build Network* procedure guarantees to find the augmented naive Bayes model with $e\le n-1$ edges that maximises this term, and so maximises the log likelihood.

□

Finally, the *Build Network* procedure minimises the upper bound of the Bayes probability of error $P(error)$ (Hellman and Raviv, 1970).

Theorem III: Let $\mathbf{D}$ be a collection of N instances of $X_1,\ldots,X_n,C$ which are *i.i.d.* Amongst all the network structures with $e\le n-1$ edges, the procedure *Build Network* generates an augmented naive Bayes classifier that minimises the upper bound of the Bayes probability of error.

Proof: Hellman and Raviv (1970) proved that the Bayes probability of error has the following upper bound,

$$P(error) \leq \frac{1}{2} H(C \mid \mathbf{Z}), \tag{3.16}$$

where C is the class variable and $\mathbf{Z} = \{X_1, \ldots, X_n\}$ is a set of n discrete-valued features. Using the following property: $H(\mathbf{Y} \mid \mathbf{X}) = -H(\mathbf{X}) + H(\mathbf{X};\mathbf{Y})$, and let $\mathbf{X} = \{X_1, \ldots, X_n, C\}$ and by using (2.2) to approximate the joint probability distribution, then

$$\begin{aligned} H(C \mid \mathbf{Z}) &= -H(\mathbf{Z}) - \sum_{\mathbf{X}} P(\mathbf{X}) \log P(\mathbf{X}) \\ &= -H(\mathbf{Z}) - \sum_{\mathbf{X}} P(\mathbf{X}) \sum_{i=1}^{n} \log P(X_i \mid \Pi_{X_i}). \end{aligned} \tag{3.17}$$

Now, by adapting (3.17) to the augmented naive Bayes with $e \leq n-1$ edges model, as done in the previous demonstrations, and following similar procedures carried out in the proof of Theorem I, this results in

$$\begin{aligned} H(C \mid \mathbf{Z}) = -H(\mathbf{Z}) - \sum_{i,\pi(i)>0} I(X_i; X_{\pi(i)} \mid C) - \sum_{i,\pi(i)=0} I(X_i;C) + \sum_{i,\pi(i)>0} H(X_i \mid C) \\ + \sum_{i,\pi(i)=0} H(X_i), \end{aligned} \tag{3.18}$$

and using the following property: $I(\mathbf{X};\mathbf{Y}) = H(\mathbf{X}) - H(\mathbf{X} \mid \mathbf{Y})$, (3.18) becomes

$$H(C \mid \mathbf{Z}) = -H(\mathbf{Z}) - \sum_{i,\pi(i)>0} I(X_i; X_{\pi(i)} \mid C) - \sum_{i} I(X_i;C) + \sum_{i} H(X_i). \tag{3.19}$$

Since $I(X_i;C)$, $H(X_i)$, and $H(\mathbf{Z})$ for all i are independent of the resulting network structure and $I(X_i; X_{\pi(i)} \mid C)$ is non-negative, minimising $H(C \mid \mathbf{Z})$ is equivalent to maximising the term

$$\sum_{i,\pi(i)>0} I(X_i; X_{\pi(i)} \mid C).$$

The *Build Network* procedure guarantees to find the augmented naive Bayes model with $e \leq n-1$ edges that maximises this term, thereby minimising the upper bound of the Bayes probability of error.

□

These three theorems show that the resulting network structure is optimum in the joint probability approximation, the log-likelihood, and the Bayes probability of error for the $e \leq n-1$ edges class network. The following section describes the methods used to test the new Bayesian network classifier.

3.4 Methods

To explore the potential of the proposed simple Bayesian network classifier as well as how the amount of training data influences the complexity (number of edges) of the network, tests were performed on 10 data sets listed in Table 3.1. These benchmark data sets come from the UCI machine learning repository (Murphy and Aha), with the exception of "corral" which was designed by John and Kohavi (1997) for feature selection experiments. For the moment, missing values are not addressed in this work, so all the instances that have one or more attribute value missing were removed. Also, a pre-discretisation step was applied to continuous attributes using Fayyad and Irani (1993) discretisation method, since the handling of continuous attributes was not addressed at the time. The accuracy of each classifier was measured by the percentage of correct predictions on the test sets of each data set.

For the first experiment, a comparison of the proposed method with the naive Bayesian classifier (**NB**) and the tree augmented naive Bayes (**TAN**) was carried out. Each data set was split randomly in to two subsets, one for training and the other for testing. Although the actual values of the percentage of correct classification will be biased depending on what data examples ended in the training set and the testing set, the main purpose of this experiment was to show how the amount of training data influences the number of augmenting edges when using the proposed incremental learning method and how its classification performance compared to that of the other two classifiers using the same training set.

The second experiment was a real industrial application. It entailed the recognition of control chart patterns. Control charts are used for displaying and monitoring variations in a process (Grant and Leavenworth, 1988). In addition to using control rules on the charts to find abnormal conditions after they have occurred, the quality control specialist also monitors control charts for long term patterns in order to predict potential problems. In this experiment, six classes of patterns were predicted: normal patterns, cyclic patterns, patterns exhibiting increasing trends, patterns showing decreasing tends, patterns with an upward shift and patterns with a downward shift. Using the expression presented in Pham and Oztemel (1992), 1500 patterns (250 for each class) were generated. The discrete time at which each pattern was sampled was taken as being within the range 0 to 59. This means there were 60 attributes. Since the main purpose of this experiment was to evaluate the classification performance of the proposed method, accuracy was measured using five-fold *cross validation* as in Kohavi (1995), thereby reducing the bias of having only one set for training and one set for testing. In addition, the **C4.5** decision-tree induction method developed by Quinlan (1993) and a multi-layer perceptron (**MLP**) were employed for comparison purposes. The MLP was trained using standard back-propagation, with 20 neurons in the hidden layer. The learning rate and the momentum coefficient were equal to 0.3 and 0.9 respectively. The same cross-validation folds were used in the three classifiers.

Table 3.1 Description of the data sets used in the first experiment

Dataset	# Attributes	# Classes	#Instances	
			Train	Test
Cleve	13	2	148	148
Corral	6	2	64	64
Flare	10	2	700	366
Heart	13	2	135	135
Hepatitis	19	2	40	40
Ionosphere	34	2	176	175
Iris	4	3	75	75
Liver-Disorders	6	2	100	245
Pima	8	2	384	384
Zoo	16	7	51	50

3.5 Experimental results

The results of the first experiment are shown in Table 3.2. In order to see how the amount of data used for training influences the complexity of the network when an incremental learning approach described in this work is used, the "Cleve" data set is analysed in detail. First, 10% of the entire data set (296 instances) was used for training. This resulted in the NB model shown in Fig. 3.4. Then if 80% of the data set is employed for training, the resulting structure is that of the TAN model with twelve augmenting edges, illustrated in Fig. 3.5. The edges from the class node C to each attribute are not considered in the edge count since it is the same for all the models. Finally if 50 % of the data is used (as described in Table I), the number of augmenting edges is 6, as shown in Fig. 3.6. This number of edges is due to the K_e values illustrated in Fig. 3.7, the sixth edge being the last, with a negative value of K_e. If an NB and a TAN model are trained using half of the data set, the network with six edges constructed using the incremental learning outperforms the other two. The proposed simple Bayesian network (SBN) classifier outperforms the other two classifiers in three more data sets. Also it is worth noticing that the proposed method can result in an NB or TAN model depending on the data used for training, but in those cases, either the NB or TAN model are the ones that have the best performances. In some cases, the accuracy is equal to that of the NB model but employing some edges and equal to that of the TAN model using fewer edges.

The control charts pattern recognition results are illustrated in Table 3.3. Each validation performed in the five-fold cross validation is shown. The simple Bayesian network classifier outperforms C4.5 and is competitive with the MLP. Notice that the resulting structure in each one of the validations is a network with less edges than the TAN model, two cases been the NB model. The resulting network for the second fold is shown in Fig. 3.8.

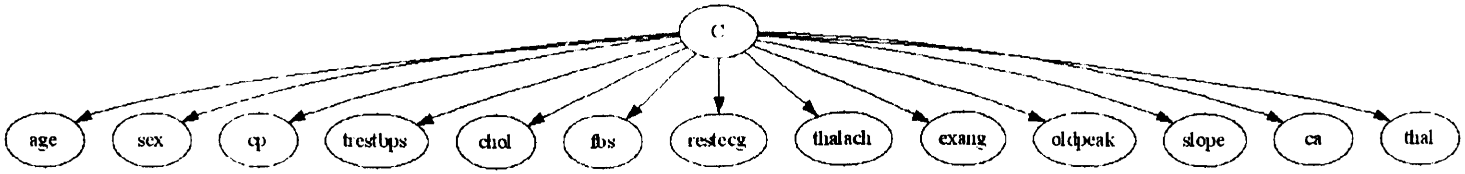

Figure 3.4 Naive Bayes model for the “Cleve” data set obtained by the SBN classifier when 10% of the instances are used for training.

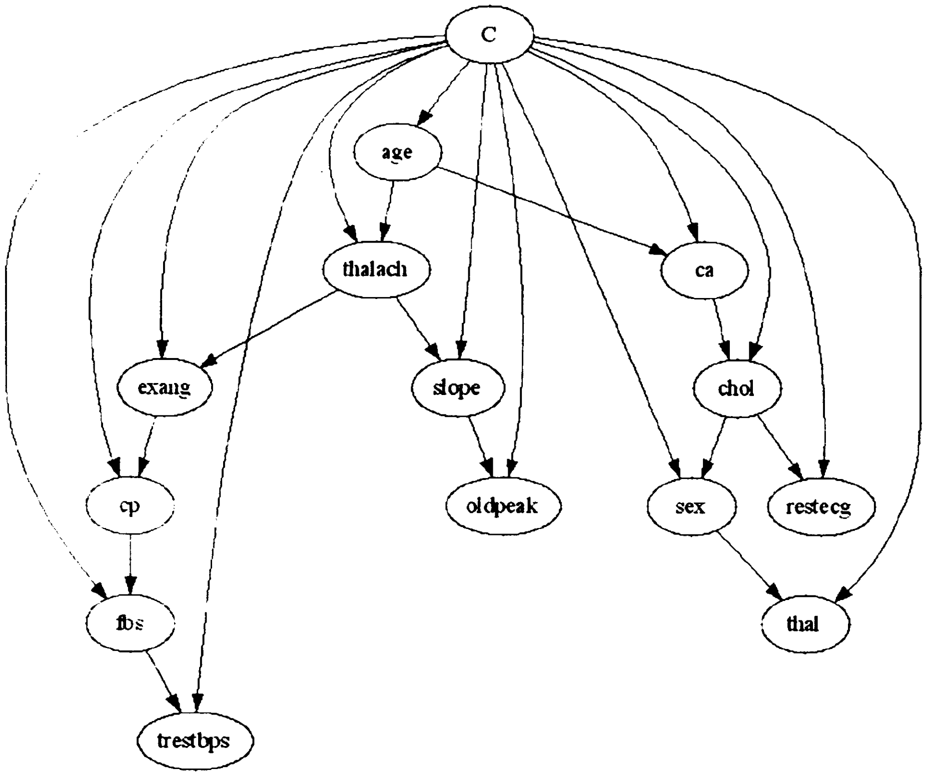

Figure 3.5 TAN model for the "Cleve" data set obtained by SBN classifier when 80% of the instances are used for training.

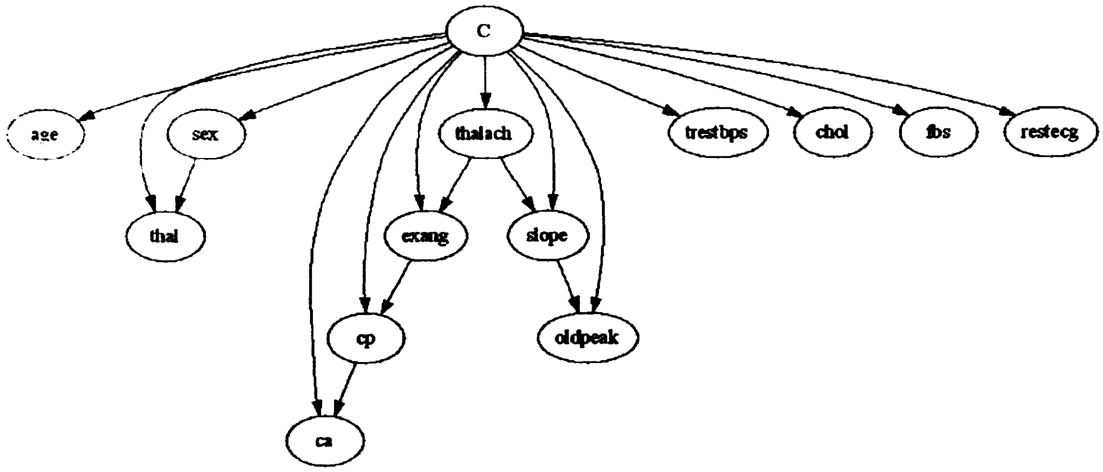

Figure 3.6 Network structure for the "Cleve" data set obtained by the SBN classifier when 50% of the instances are used for training.

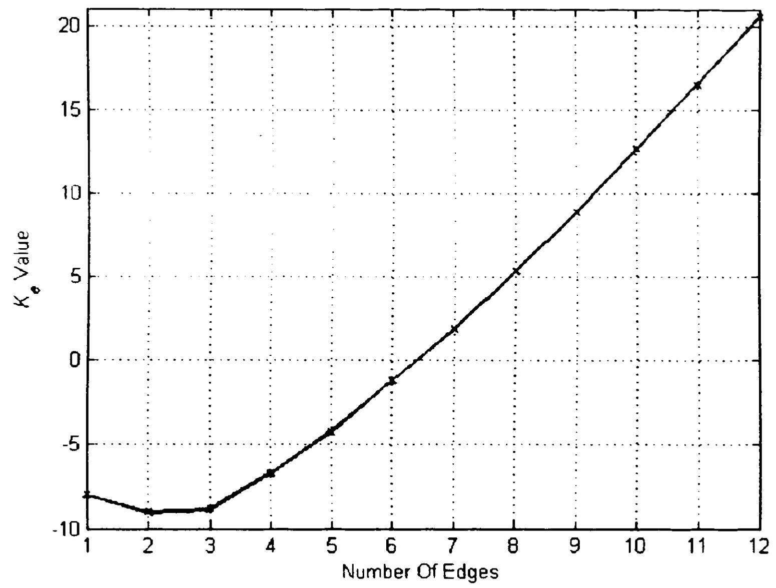

Figure 3.7 Plot illustrating the K_e values for each edge added during the Build Network procedure for the "Cleve" data set, when 50% of the instances for training are used.

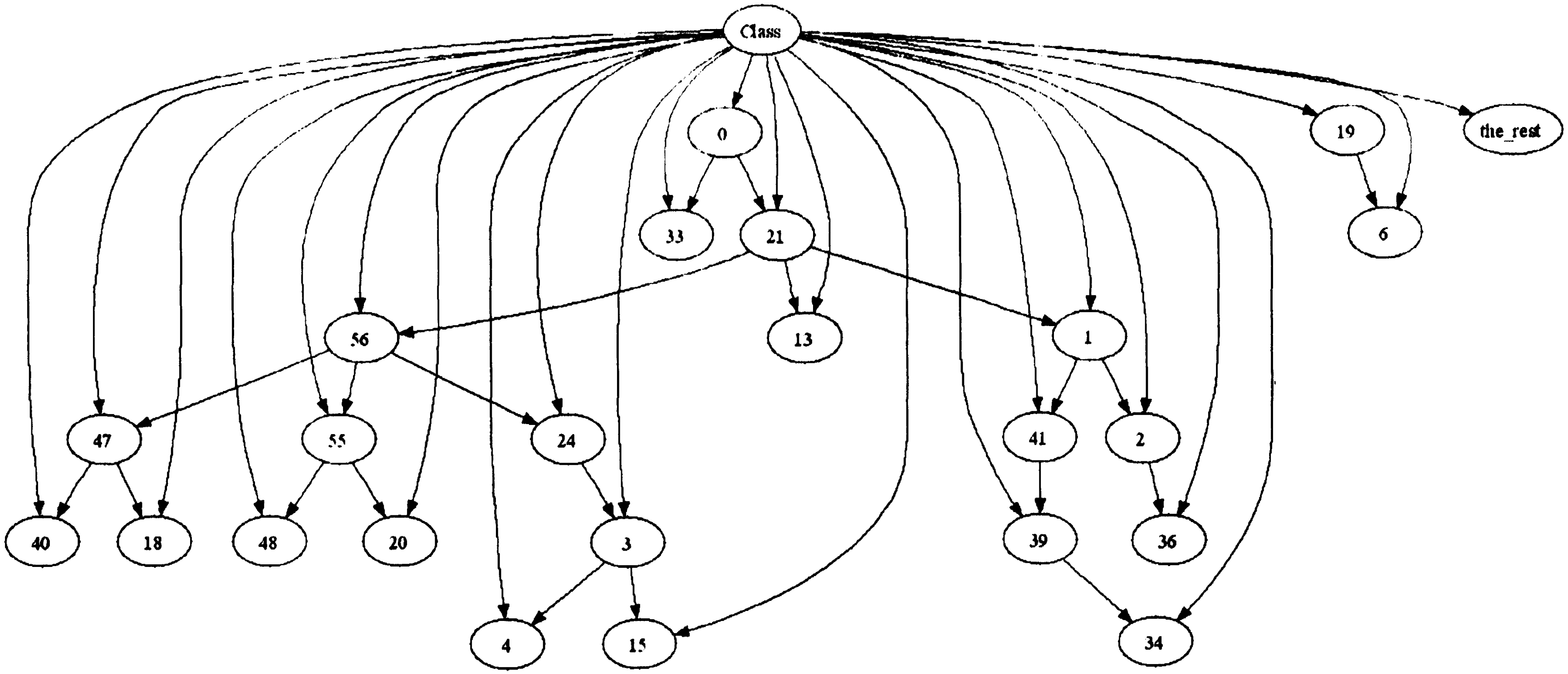

Figure 3.8 The simple Bayesian network classifier for the control chart data

Table 3.2 Comparison of the NB, TAN, and the SBN on the test sets

Dataset	NB	TAN	SBN (e / n-1 edges)
Cleve	79.73	81.76	83.12 (6 / 12)
Corral	82.81	93.75	100.00 (3 / 5)
Flare	80.60	84.15	84.15 (9 / 9)
Heart	84.44	84.44	85.19 (9 / 12)
Hepatitis	95.00	72.50	95.00 (4 / 18)
Ionosphere	93.14	93.71	93.71 (12 / 33)
Iris	97.33	97.33	97.33 (0 / 3)
Liver-Disorders	71.43	69.80	71.84 (3 / 5)
Pima	77.08	77.60	77.60 (7 / 7)
Zoo	88.00	78.00	88.00 (2 / 15)

Table 3.3 Classification accuracy on the control chart pattern recognition problem

CV-folds	C4.5	MLP	SBN (e / n-1) edges
1st	90.33	98.33	99.33 (0 / 59)
2nd	89.00	97.66	99.33 (21 / 59)
3rd	92.00	99.00	99.33 (0 / 59)
4th	90.66	97.33	98.00 (23 / 59)
5th	93.33	98.66	98.66 (13 / 59)
Mean	91.06	98.19	98.93

3.6 Summary

In this chapter, a new method for augmenting the naive Bayesian classifier in an incremental way is presented. The complexity of a Bayesian network (the number of augmenting edges) is incorporated in the learning process by the use of the Bayes factor. Although there are no universally accepted ways of assigning the prior probabilities that the Bayes' theorem requires, in this work coding theory was used to assign these probabilities following an Occam's razor approach, that is, networks with less edges have higher prior probabilities than more complex ones. Using Chow and Liu's dependence tree algorithm, as done by the Tree Augmenting Naive Bayes (TAN) classifier, an incremental learning method is developed which incorporates the complexity of the network yielding structures that vary from no edges (naive Bayesian classifier) to $n-1$ edges (TAN model) where n is the number of attribute nodes. The resulting Bayesian network structure learned by the incremental approach augments the naive Bayesian classifier with $e \leq n-1$ edges that are optimum in the sense that they minimise the cross-entropy between the real joint probability distribution and the approximation done by the Bayesian network, as well as the Bayes probability of error and maximise the log-likelihood between the Bayesian network and the data. From the experimental results, the following conclusions can be drawn.

First, the complexity of the Bayesian network classifiers depends directly on the amount of data used for training. A spanning tree structure like the TAN model is therefore only justified when there is sufficient data. This is not a problem for the incremental learning method since it automatically regulates the number of edges permitted depending on the amount of training data.

Second, the classification performance of the proposed incremental learning approach when compared to that of the TAN and the naive Bayesian models is usually better or at least equal to them. When the TAN model obtains the best accuracy, the proposed method can either end up with a TAN model structure as well or sometimes can reach the same classification performance with fewer edges. On the other hand, when the naive Bayes model is the one with the best results, the proposed method can sometimes reach the same accuracy with a few augmenting edges. This is favourable when some type of probabilistic dependence knowledge amongst the attributes is required, as well as having the best classification performance. In these cases, if a TAN model is used, although more complete dependence knowledge can be obtained, the classification performance is lower.

Third, the classification performance of the Bayesian network classifier, compared to non-Bayesian classifiers using real-world problem data, outperformed the C4.5 and demonstrated to be competitive to a neural network, obtaining near to 99% in correct classification. Also, it can be pointed out that for each one of the folds in the five-fold cross validation experiment, the Bayesian network classifier presents less variability than the neural network. This can be explained, in part, due to the limited amount of edges in the network structure.

In general, the SBN classifier with its incremental learning method tries to overcome the bias/variance dilemma also known as *overfitting*, thereby improving the generalisation power. Future work will be concentrated on allowing the use of continuous value attributes. An extension to this work for unsupervised learning to deal with clustering tasks will be described in the next chapter.

Chapter 4

UNSUPERVISED TRAINING OF BAYESIAN NETWORKS

4.1 Preliminaries

The learning of Bayesian network classifiers from data is commonly performed in a *supervised* manner, meaning that a training set containing examples which have been previously classified by an expert are used to generate the DAG and its CPT. In practice, this can be viewed as having a class label assigned to each example (row) of the data set. Unfortunately, in many real-world applications of machine learning in industrial problems, it is difficult to obtain a large data set with classified examples. This is generally due to the fact that a human expert is needed to manually classify each example, of which in many cases there can be thousands, making it an exhausting and time consuming task. With this in mind, it is desirable to have an alternative way of training a classifier with data that has no class label assigned to each example. This approach is known as *unsupervised training*, also referred to as *clustering*. The main goal of clustering is to find the natural groupings of the data. Well known clustering algorithms are: the k-means (MacQueen, 1967), the Fuzzy C-Means (a fuzzy version of k-means) (Bezdek, 1981), information theory based clustering (Roberts *et al.*, 2000), and the Expectation-Maximization (EM) algorithm (Dempster *et al.*, 1977). It is important to mention at this point that unsupervised training will usually obtain lower performances when compared to models trained in a supervised way, primarily due to the lack of the class information during the learning process. However, it is still a very valuable tool for data exploration and preliminary classification, which can be improved later on once a training set with class labels for each example is built. This task can be carried out by a human expert with assistance of the clustering results, making it a less difficult task.

Although Bayesian network classifiers have received a lot of attention recently, most of the efforts have been concentrated in developing supervised learning algorithms. Less work has been reported on the unsupervised training of Bayesian network classifiers. In the following section, previous work done on the unsupervised training of Bayesian network classifiers is discussed.

In Barash and Friedman (2002) a clustering technique called *context-specific independences* (CSI) is used for clustering genes based on a combination of genomic and genetic data. The CSI is a refinement of the selective Bayesian models, which are essentially a special subclass of Bayesian networks, specifically, a naive Bayes classifier with the difference that not all the attributes are class-dependant. Learning for these models is performed by the *Bayesian structural EM algorithm* (Friedman, 1998), which is capable of learning parameters and structures in an unsupervised way using a scoring function such as the *Bayesian Information Criterion* (BIC) or the *Cheeseman-Stutz* (CS) (Neapolitan, 2004).

An extension of model averaging (MA) with naive Bayesian classifiers (Dash and Cooper, 2002) for clustering problems is presented in Santafé *et al.* (2006a). To accomplish this, the expectation MA (EMA) algorithm is introduced, which incorporates the MA calculations in the maximisation step of the EM algorithm. Tests carried out on synthetic data and DNA microarray data show that the EMA algorithm is a powerful learning algorithm that can be useful for clustering problems where there are many attributes and only a few examples. An extension made to the EMA algorithm to learn TAN models is presented in Santafé *et al.* (2006b).

A heuristic algorithm for learning Bayesian networks for clustering is shown in Peña *et al.* (1999). The approach is based upon improving the naive Bayes model by means of constructive induction, which is the process of changing the representation of the examples in the database by creating new attributes from existing attributes. With this, some violation of conditional independence assumptions made by the NB model are detected and dependencies among attributes are included in the model. The parameter search is performed either by the standard EM algorithm, or a hybridisation of the Bound and Collapse method and the EM algorithm (BC+EM), which results in a technique that exhibits a faster convergence rate and a more effective behaviour than the EM algorithm. In Peña *et al.* (2000), the BC+EM method is also used to improve the Bayesian structural EM algorithm (BSEM) for learning Bayesian networks for clustering.

Estimation of distribution algorithms for the unsupervised training of Bayesian networks, both directly and within the framework of the BSEM algorithm, is proposed and empirically evaluated in Peña *et al.* (2004). An application of the proposed method to gene expression data clustering showed that the identified clusters, after being validated, may be biologically meaningful.

In this chapter, an unsupervised training approach following the classification EM algorithm (CEM) (Celeux and Govaert, 1992) framework is developed for three types of Bayesian network classifiers: Chow & Lui multinets, the TAN model, and the simple Bayesian network classifier which is more robust in its structure, capable of handling the trade-off between complexity (number of edges in the network) and accuracy using a Bayesian approach. The unsupervised training technique maximises the classification maximum likelihood (CML) of the Bayesian network classifiers, instead of using the traditional EM approach which maximises the maximum likelihood (ML). Results on a benchmark data set and on a real-world industrial problem, the clustering of wood defects, are presented together with a comparison with traditional clustering algorithms.

The chapter is organised as follows: section 4.2 presents a description of probabilistic classifiers. The description of the CEM algorithm is briefly presented in section 4.3. The proposed unsupervised training via the CEM framework is developed in section 4.4. A description of the experiments carried out to test the proposed method as well as the description of the application to an industrial problem is presented in section 4.5. The results are illustrated in section 4.6 as well as the discussions. The general summary of the proposed method and future works are analysed in section 4.7.

4.2 Probabilistic classifiers

In classification tasks, the idea in probabilistic classification is to find the class value which maximises the posterior probability of the class for a given set of assignments to the attributes. In other words, the class value for the r^{th} example of the data set, $X_1 = x_1^r, X_2 = x_2^r, \ldots, X_n = x_n^r$, can be obtained as:

$$\begin{aligned} &class\_value(X_1 = x_1^r, \ldots, X_n = x_n^r) \\ &\qquad = \arg\max_k P(C = k \mid X_1 = x_1^r, \ldots, X_n = x_n^r), \end{aligned} \tag{4.1}$$

and by using Bayes' theorem, the posterior probability can be computed as

$$P(C \mid X_1, \ldots, X_n) = \frac{P(C)P(X_1, \ldots, X_n \mid C)}{P(X_1, \ldots, X_n)}. \tag{4.2}$$

On the r.h.s. of Eq. (4.2) the denominator is constant with respect to the class, and can be expressed as $1/\beta$. So, the main challenge is how to compute the numerator. The simplest approach is to assume that each attribute is conditionally independent of every other attribute. This rather "naive" assumption yields the well known naive Bayesian classifier,

$$P(C \mid X_1, \ldots, X_n) = \beta P(C) \prod_{i=1}^{n} P(X_i \mid C). \tag{4.3}$$

Equation (4.3) has a Bayesian network representation since it is a special case of Eq.(2.2), where the class node C is the vertex and there is an edge (arc) from C to each attribute X_i, as can be seen in Fig. 4.1.

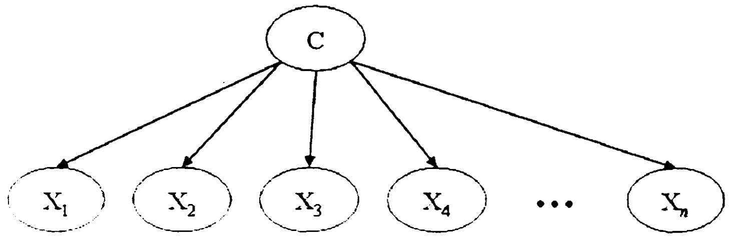

Figure 4.1 Bayesian network representation of the naive Bayes classifier

Another approach is to use Chow & Liu's tree algorithm. In this case, the training set needs to be partitioned into groups that have examples belonging to the same class. Then, for each one of these groups a Bayesian network is learned, with the restriction that each attribute has only, at most, one other attribute as a parent yielding tree structures that have n-1 edges. The learning algorithm for this type of Bayesian network is shown in Fig. 4.2. The initial step is to compute the *mutual information* between each pair of attributes, defined as

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}. \tag{4.4}$$

Basically, this function measures the amount of information that Y provides about X. It is important to point out that in this work the marginal and conditional probabilities are estimated by the empirical frequencies from the data.

The following step is to build a complete undirected graph. This is carried out by connecting an edge from each node (attribute) to every other node and assigning the weight of the edge that connects X_i with X_j by $I(X_i;X_j)$. Next, in order to obtain a tree structure, the *maximum weighted spanning tree* (MWST) is built using any well known MWST procedures, such as Kruskal's algorithm (Kruskal, 1956). Finally, directions to the edges of the resulting tree can be added by choosing any attribute as the root and then setting the directions of all edges to be pointing outwards from it.

If there are K classes then K CL trees are learned, one for each group of data (with the same class value). Each tree distribution P_k, with $k = 1 \ldots K$, will be approximating the joint probability distribution of the attributes, given a specific class,

$P(X_1,\ldots,X_n \mid C=k)=P_k(X_1,\ldots,X_n)$. Since CL trees are Bayesian networks, the joint probability distribution can be computed by Eq. (2.2), where $\Pi_{X_i}=\{X_{j\neq i}\}$ will only have one attribute, except for the root attribute node which will have no parents $\Pi_{X_{root}}=\{\varnothing\}$. Then the CL multinet classifier can be expressed as

$$P(C=k \mid X_1,\ldots,X_n)=\beta P(C=k)\prod_{i=1}^{n} P_k(X_i \mid \Pi_{X_i}), \qquad (4.5)$$

and a representation can be seen in Fig. 4.3.

The following classifier to be analysed is the TAN classifier. This model aims to overcome the strong independence assumption that the naive Bayesian classifier imposes amongst the attributes in order to obtain Eq. (4.3). The improvement is accomplished by augmenting the naive Bayes model with edges via a modification of the CL tree algorithm. A general outline of the TAN learning process is shown in Fig. 4.4. The main difference with the CL tree procedure is that the TAN model uses the *conditional mutual information*, defined as

$$I(X;Y \mid Z)=\sum_{x\in X}\sum_{y\in Y}\sum_{z\in Z} P(x,y,z)\log\frac{P(x,y \mid z)}{P(x \mid z)P(y \mid z)}. \qquad (4.6)$$

The conditional mutual information measures the information that Y provides about X when the value of Z is known. By using this function, a unique tree structure is obtained instead of the CL multinet which builds a tree for each class.

In this model, each attribute will have $\Pi_{X_i}=\{X_{j\neq i},C\}$, except for the root attribute node which will have $\Pi_{X_r}=\{C\}$. Then, the TAN classifier can be expressed as

$$P(C \mid X_1,\ldots,X_n) = \beta P(C)\prod_{i=1}^{n} P(X_i \mid \Pi_{X_i}), \tag{4.7}$$

and its Bayesian network representation appears in Fig. 4.5.

The last model described in this section is a new Bayesian network classifier introduced in the previous chapter which follows the TAN model procedure but it is not restricted to tree structures (n-1 edges). In fact, the complexity of the network (number of edges) is automatically regulated during the training process, obtaining structures that can have a number of edges e that range from 0 (naive Bayes model) up to n-1 (TAN model). The key difference with the standard TAN procedure is that in step three (see Fig. 4.4), in each iteration of the MWST, a Bayesian measure with the selected edge to be added must be computed and checked if the stopping criterion for the MWST algorithm is met. If the stopping criterion is met before the MWST algorithm adds the final edge, then the resulting structure will have $e < n-1$ edges. The Bayesian measure that is computed takes in to account the number of training examples. By doing this, it automatically regulates the complexity of the structure, allowing a resulting structure like the TAN model to exist only when there is sufficient information (data) to support $n-1$ edges. The Bayesian measure λ that needs to be computed every time an edge is selected, for example an edge from attribute X_ω to attribute X_υ, for building the maximum weighted spanning tree, is

$$\lambda = 2m + \sum_{r=1}^{N} \log_2 P(x_\upsilon^r \mid c^r) - \log_2 P(x_\upsilon^r \mid x_\omega^r, c^r), \tag{4.8}$$

where N is the number of training examples or instances in the training data set and $m = \log_2(n+1)$. The derivation of Eq.(4.8) appears in chapter 3.

The stopping criterion for the MWST algorithm is to check that the accumulated value of λ in each iteration is negative, in other words, for the e^{th} edge to be added it must satisfy

$$\Lambda_e = \sum_{i=1}^{e} \lambda_i < 0, \tag{4.9}$$

where λ_i represents the λ value for the i^{th} edge being considered (please see chapter 3 for a more detailed explanation of this criterion).

The general learning procedure of this model called the simple Bayesian network (SBN) classifier appears in Fig. 4.6. Although this model uses Eq. (4.7) to carry out the classification, it is worthwhile pointing out that the structure of the network may not be a tree. Thus, if the resulting structure has $e < n - 1$ edges, then there are only e attributes with $\Pi_{X_i} = \{X_{j \neq i}, C\}$ and all the rest of the attributes will have $\Pi_{X_i} = \{C\}$, including the root attribute. An illustration of this type of Bayesian network classifier is shown in Fig. 4.7.

For the three Bayesian network classifiers described in this section, the learning procedure for each one has time complexity $O(n^2N)$, associated with the first step (computation of the mutual information/conditional mutual information between each pair of attributes) of the learning procedure. For more details see Friedman *et al.* (1997).

CL tree procedure

1. Compute the mutual information *$I(X_i;X_j)$ between each pair of attributes $i \neq j$.*
2. Build a complete undirected graph using the attributes as nodes and assign the weight of the edge that connects X_i to X_j by $I(X_i;X_j)$.
3. Apply the MWST algorithm.
4. Choose an attribute to be root and set the directions of all the edges to be outward from it.

Figure 4.2 Chow & Liu's tree algorithm

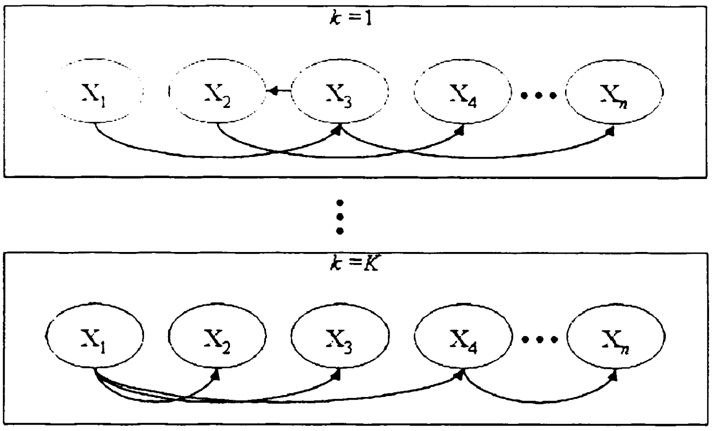

Figure 4.3 CL multinet classifier representation

TAN procedure

1. Compute the conditional mutual information $I(X_i;X_j|C)$ *between each pair of attributes* $i \neq j$.
2. Build a complete undirected graph using the attributes as nodes and assign the weight of the edge that connects X_i to X_j by $I(X_i;X_j|C)$.
3. Apply the MWST algorithm.
4. Choose an attribute to be root and set the directions of all the edges to be outward from it.
5. Add a vertex node C and add an edge from C to every other attribute X_i.

Figure 4.4 TAN learning process

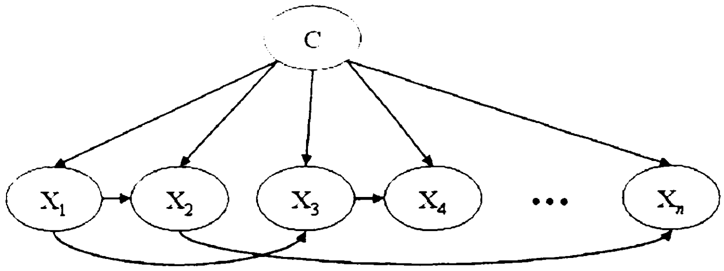

Figure 4.5 TAN classifier representation

SBN classifier procedure

1. Compute the conditional mutual information $I(X_i;X_j|C)$ *between each pair of attributes* $i \neq j$.
2. Build a complete undirected graph using the attributes as nodes and assign the weight of the edge that connects X_i to X_j by $I(X_i;X_j|C)$.
3. Apply the MWST algorithm. For each iteration of the MWST algorithm:
   3.1 Compute the Bayesian measure λ from Eq.(4.8).
   3.2 Check that Λ satisfies condition in Eq.(4.9).
4. Choose an attribute to be root and set the directions of all the edges to be outward from it.
5. Add a vertex node C and add an edge from C to everv other attribute X_i.

Figure 4.6 SBN classifier learning procedure

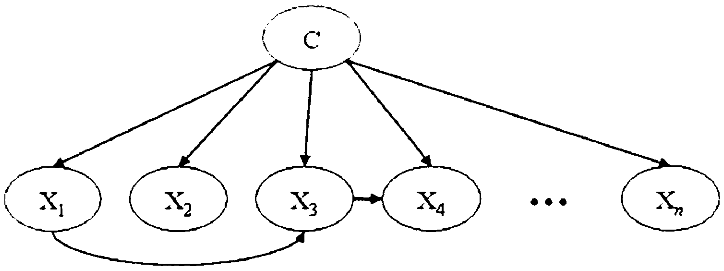

Figure 4.7 SBN classifier representation

4.3 Classification EM algorithm

In unsupervised training the class attribute is latent or in Bayesian network terminology *hidden*. The aim of the proposed unsupervised training method is to be able to build a Bayesian network classifier and assign "ideally" the correct cluster (class) value to the latent class variable of the data. In order to develop the learning process, the Classification EM algorithm (Celeux and Govaert, 1992) framework will be used. To follow this approach, the data will be assumed to have been sampled from a mixture model, where the number of the mixing component distributions will be the same as the number of clusters trying to be discovered. Then, a mixture model for a set of n discrete random attributes $\{X_1,\dots X_n\}$ with K mixing components, can be expressed as

$$P(\mathbf{X}) = \sum_{k=1}^{K} \alpha_k f_k(\mathbf{X}) \tag{4.10}$$

$$\sum_{k=1}^{K} \alpha_k = 1, \qquad \alpha_k \geq 0$$

where the α_k are the mixing coefficients (weights) and the f_k are the mixing component distributions.

The mixture parameters θ (mixing coefficients as well as the parameters of the mixing distributions) can be obtained by two frequently used maximum likelihood approaches: the mixture approach and the classification approach (Celeux and Govaert, 1995). Roughly speaking, the mixture approach aims to maximise the likelihood over the mixture parameters. On the other hand, the classification approach aims to maximise the likelihood over the mixture parameters, but also, over the identifying labels of the mixture component origin for each data example, i.e., to what cluster does each data example belong. In the mixture approach, the parameter θ is chosen to maximise the log-likelihood using the EM algorithm. Since the aim of this work is to build Bayesian network classifiers (to find the structure and network parameters) as well as to cluster the data examples, the classification approach is more appropriate. In this approach, the indicator vectors, identifying the mixture component origin, $z^r = (z_k^r, k = 1,\ldots,K)$ with $z_k^r = 1$ or 0 if $\mathbf{x}^r$ $(1 \le r \le N)$ has been drawn from the k^{th} component or not, are treated as unknown parameters. Then the parameter θ and the indicator vectors z^r are chosen to maximise the Classification Maximum Likelihood (CML) criteria, which under the mixture sampling scheme is defined as (Celeux and Govaert, 1995)

$$CL(\theta, z^1,\ldots,z^N \mid \mathbf{x}^1,\ldots,\mathbf{x}^N) = \sum_{k=1}^{K} \sum_{\mathbf{x}^r \in \mathrm{P}_k} \log f_k(\mathbf{x}^r) + \sum_{k=1}^{K} n_k \log \alpha_k, \qquad (4.11)$$

where $\mathrm{P} = \{\mathrm{P}_1,\ldots,\mathrm{P}_k\}$ is a partition (cluster) of the N data examples $\mathbf{x}^1,\ldots,\mathbf{x}^N$ associated to the indicator vectors $z^1,\ldots,z^N : \mathrm{P}_k = (\mathbf{x}^r \mid z_k^r = 1)$ and $n_k = \# \mathrm{P}_k$ $(1 \le k \le K)$. Equation (4.11) can be optimised by a classification version of the EM algorithm called the Classification EM (CEM) algorithm (Celeux and Govaert, 1992). The CEM algorithm incorporates a classification step between the E-step and the M-step of the EM algorithm using a *maximum a posteriori* (MAP) principle. A description of the algorithm follows.

Starting from an initial partition P^0, the m^{th} $(m > 0)$ iteration of the CEM algorithm consists in:

E-step: compute for $r = 1,\ldots,N$ and $k = 1,\ldots,K$ the current posterior probabilities that $\mathbf{x}^r$ belongs to P_k.

C-step: update the partition by assigning each $\mathbf{x}^r$ to the cluster which provides the maximum posterior probability. Let P^m denote the new partition.

M-step: for $k = 1,\ldots,K$ compute the maximum likelihood estimates of θ^m using the cluster P_k^m as sub-samples.

The theoretical properties of the CEM algorithm, such as the convergence of the algorithm, are described in (Celeux and Govaert, 1992).

The next section will show how to learn Bayesian network classifiers under the CML criteria via the CEM algorithm.

4.4 Proposed unsupervised training approach

Unsupervised training of the CL multinets, TAN, and the SBN classifier described in section 4.2 will be carried out assuming that the data examples have been generated from a mixture of Bayesian networks. This mixture can be described by Eq. (4.10), resulting in a mixtures of trees in the case of CL multinets, a mixtures of trees with shared structures in the case of the TAN model, and a mixture of networks with shared structures that can vary from the naive Bayes model to the TAN model in the case of the simple Bayesian network classifier. Although CL trees are Bayesian networks, a mixture model of them is not a Bayesian network, whereas by learning networks with the same structure, the resulting mixture model is a unique Bayesian network, which is the case of the TAN model and the SBN classifier. Learning with mixtures of trees for density estimation and classification tasks using the EM algorithm has been developed in Meilă and Jordan (2000), but not for clustering tasks as in this work. As mentioned before, this work's interest is to learn the Bayesian network classifiers, as well as to cluster the data examples. For this, the mixture parameters will be obtained following the classification approach which maximises Eq.(4.11) using the CEM algorithm. This will be developed in what follows.

4.4.1 Unsupervised training for CL multinets classifier

Starting from an initial partition P^0, the E-step in the m^{th} iteration for $r = 1,\dots,N$ and $k = 1,\dots,K$, consists in estimating the probability of each tree generating data point $\mathbf{x}^r$, i.e., the posterior probability that $\mathbf{x}^r$ belongs to P$_k$. This posterior probability is computed by

$$t_k^m(\mathbf{x}^r) = \frac{\alpha_k^m \prod_{i=1}^{n} P_k^m(x_i^r \mid \Pi_{x_i^r})}{\sum_{k'=1}^{K} \alpha_{k'}^m \prod_{i=1}^{n} P_{k'}^m(x_i^r \mid \Pi_{x_i^r})}, \tag{4.12}$$

where $\Pi_{x_i^r}$ is the value of Π_{X_i} in the r^{th} example. If $X_{j(i)}$, with $j \neq i$, is the parent of X_i then $\Pi_{x_i^r} = x_{j(i)}^r$ in Eq. (4.12). Also note that the α_k are in fact the probability distribution of the variable class C, i.e., $P(C = k)$. It is important not to confuse the notation P_k , the probability distribution of the k^{th} tree, with P$_k$ which is the k^{th} partition (cluster). In the C-step each $\mathbf{x}^r$ is assigned to the cluster k that provides the maximum posteriori probability Eq. (4.12), $1 \leq k \leq K$. If the maximum posteriori probability is not unique, assign $\mathbf{x}^r$ to the cluster with the smallest index. Let the resulting partition be denoted by P^m. The initial iterations of the CEM algorithm may not be so reliable due to the dependence on the initial partition, causing a convergence to local optima of the CML function. In Celeux and Govaert (1992) a way of reducing this problem was shown by replacing the C-step by a stochastic step called the S-step. In the S-step each $\mathbf{x}^r$ is assigned at random to one of the clusters $\mathrm{P}_1,\dots,\mathrm{P}_K$ with probability $t_k^m(\mathbf{x}^r), k = 1,\ldots,K$. The resulting partition is denoted by P^m. The idea in general is to start of by using the S-step up to a certain number of iterations, defined by the user, and then to swap the S-step for the C-step for the final iterations. The M-step for $k = 1,\ldots,K$ consist in maximising the CML criteria using the sub-samples P_k^m. Equation (4.11) under the mixtures of CL trees can be expressed as

$$CL = \sum_{k=1}^{K} \sum_{\mathbf{x}^r \in \mathrm{P}_k} \log \prod_{i=1}^{n} P_k(x_i^r \mid \Pi_{x_i^r}) + \sum_{k=1}^{K} n_k \log \alpha_k. \qquad (4.13)$$

By inspecting Eq. (4.13) it can be seen that the mixture parameters are disjoined, so each sum on the right-hand side of Eq. (4.13) can be maximised independently with respect to the part of the model on which it depends. The maximisation of the second term of Eq. (4.13), taking into account the constraint

$$\sum_{k=1}^{K} \alpha_k = 1,$$

yields in

$$\alpha_k^{m+1} = \frac{n_k}{N} = \frac{\#\mathrm{P}_k^m}{N} \text{ for } k = 1,\ldots,K. \qquad (4.14)$$

The maximisation of the first term of Eq. (4.13) will result in a new tree distribution P_k^{m+1} (conditional probabilities and tree structure). To achieve this, for each k the following expression needs to be maximised

$$LL = \sum_{\mathbf{x}^r \in \mathrm{P}_k} \sum_{i=1}^{n} \log P_k(x_i^r \mid \Pi_{x_i^r}). \qquad (4.15)$$

It is recalled that there are n_k data points (examples) in partition P$_k$, so Eq. (4.15) is written as

$$LL = \sum_{r=1}^{n_k}\sum_{i=1}^{n} \log P_k(X_i = x_i^r \mid \Pi_{X_i} = \Pi_{x_i^r}), \tag{4.16}$$

with $\mathbf{x}^r$ for $r = 1,\ldots,n_k$ belonging to partition P$_k$. Equation (4.16) is the *log-likelihood* (LL) of the model (k^{th} tree) given the data and it is maximised by selecting a tree and its associated conditional probabilities. For any given tree k, Eq. (4.16) is maximised if an empirical distribution $\hat{P}$ defined by the frequencies of events in the data is used as the estimate of the conditional probabilities. This is because $\hat{P}$ is a maximum-likelihood estimator for P. By using $\hat{P}$ in Eq. (4.16), and interchanging the orders of summation, a decomposition of the log-likelihood (Friedman *et al.*,1997) according to the k^{th} tree can be expressed as

$$LL = n_k \sum_{i=1}^{n} \sum_{\substack{x_i \in X_i \\ \Pi_{x_i} \in \Pi_{X_i}}} \hat{P}_k(x_i, \Pi_{x_i}) \log \hat{P}_k(x_i \mid \Pi_{x_i}). \tag{4.17}$$

Then, with some manipulation, Eq. (4.17) becomes

$$= n_k \sum_{i=1}^{n} \sum_{\substack{x_i \in X_i \\ \Pi_{x_i} \in \Pi_{X_i}}} \hat{P}_k(x_i, \Pi_{x_i}) \log \frac{\hat{P}_k(x_i, \Pi_{x_i})}{\hat{P}_k(x_i)\hat{P}_k(\Pi_{x_i})}$$

$$+ n_k \sum_{i=1}^{n} \sum_{\substack{x_i \in X_i \\ \Pi_{x_i} \in \Pi_{X_i}}} \hat{P}_k(x_i, \Pi_{x_i}) \log \hat{P}_k(x_i)$$

$$= n_k \sum_{i=1}^{n} \hat{I}_k(x_i; \Pi_{x_i}) - n_k \sum_{i=1}^{n} \hat{H}_k(x_i), \tag{4.18}$$

where the second expression on the right-hand side of Eq. (4.18) is independent of the tree structure and $\hat{I}_k$ is the mutual information computed by the empirical distributions defined by the frequencies of events in the data of P$_k$. By defining $X_{j(i)}$, with $j \neq i$, as the parent of X_i, the tree that must be chosen in order to maximise Eq. (4.15), is the one that maximises the right-hand side of Eq. (4.18). This can be accomplished by using the MWST algorithm with $\hat{I}_k(x_i;x_{j(i)})$ as branch (edge) weights.

A summary of the unsupervised training of CL trees via the CEM algorithm is shown in Fig. 4.8.

4.4.2 Unsupervised training for the TAN classifier

In this case the tree distributions P_k for $k = 1,\ldots,K$ have the same structure. The E-step is the same as before. The posterior probabilities can be computed using Eq. (4.12). Also, the S-step and the C-step remain the same. The M-step consists in maximising the CML criteria, which has the same expression as before,

$$CL = \sum_{k=1}^{K} \sum_{\mathbf{x}^r \in \mathrm{P}_k} \log \prod_{i=1}^{n} P_k(x_i^r \mid \Pi_{x_i^r}) + \sum_{k=1}^{K} n_k \log(\alpha_k). \tag{4.19}$$

By inspecting Eq. (4.19) it can be seen that the mixture parameters are disjoined. The maximisation of the second term of Eq. (4.19) gives the same results as Eq. (4.14),

$$\alpha_k^{m+1} = \frac{n_k}{N} = \frac{\#\mathrm{P}_k^m}{N} \text{ for } k = 1,\ldots,K. \tag{4.20}$$

The maximisation of the first term is different from before since now all the trees in the mixture model have the same structure. This constraint implies that the maximisation procedure cannot now be performed separately for each one of the K trees. The maximisation needs to be done simultaneously for all the K trees.

Using the result obtained before in Eq. (4.18), the first term of Eq. (4.19) can be expressed as

$$\sum_{k=1}^{K} n_k [\sum_{i=1}^{n} \hat{I}_k(x_i;\Pi_{x_i}) - \sum_{i=1}^{n} \hat{H}_k(x_i)]$$

$$= \sum_{k=1}^{K} N\alpha_k \sum_{i=1}^{n} \hat{I}_k(x_i,\Pi_{x_i}) + \text{constant term independent of the structure.} \qquad (4.21)$$

Now, let C be defined as a variable whose values, for $r = 1,\ldots,N$, are obtained from the indicator vector as $c^r = (k \mid z_k^r = 1)$. In other words, C contains the cluster label for each data point. Also, it is recalled that the α_k are the probability distribution of the cluster/class variable C, i.e., $P(C = k)$ and that $P_k(\mathbf{X}) = P(\mathbf{X} \mid C = k)$, then Eq. (4.21) can be written as

$$= N\sum_{i=1}^{n} \sum_{\substack{x_i \in X_i \\ \Pi_{x_i} \in \Pi_{X_i} \\ c \in C}} \hat{P}(c) \frac{\hat{P}(x_i,\Pi_{x_i},c)}{\hat{P}(c)} \log \frac{\hat{P}(x_i,\Pi_{x_i} \mid c)}{\hat{P}(x_i \mid c)\hat{P}(\Pi_{x_i} \mid c)} + \text{constant}$$

$$= N\sum_{i=1}^{n} \hat{I}(x_i;\Pi_{x_i} \mid c) + \text{constant}, \qquad (4.22)$$

where the second expression on the right-hand side of Eq. (4.22) is independent of the tree structure and $\hat{I}$ is the conditional mutual information computed by the empirical distributions defined by the frequencies of events in the entire data set. By defining $X_{j(i)}$, with $j \neq i$, as the parent of X_i, the tree that must be chosen is the one that maximises the right-hand side of Eq. (4.22). This can be accomplished by using the MWST algorithm with the conditional mutual information $\hat{I}(x_i;x_{j(i)} \mid c)$ as branch (edge) weights. The learning procedure for the TAN model is summarised in Fig. 4.9.

4.4.3 Unsupervised training for the SBN classifier

This model follows the same procedure as the TAN classifier. The only difference is when maximising Eq. (4.22), since the MWST is constructed branch by branch, for every branch added it needs to check if the condition in Eq. (4.9) holds. The learning procedure for the SNB classifier is shown in Fig. 4.10.

```
CEM-CL trees
Starting from an initial partition $P^0$, the $m^{th}$
iteration of the CEM-CL trees:

E-step:     compute $t_k^m(\mathbf{x}^r)$ for $r=1,\ldots,N$ and $k=1,\ldots,K$
S/C-step:   if $m$ < user defined threshold
                 for $r=1,\ldots,N$ assign at random each $\mathbf{x}^r$
                 to one of the clusters $P_1,\ldots,P_K$ with
                 probability $t_k^m(\mathbf{x}^r)$, $k=1,\ldots,K$
            else
                 for $r=1,\ldots,N$ assign each $\mathbf{x}^r$ to the
                 cluster which provides the maximum
                 posterior probability $t_k^m(\mathbf{x}^r)$, $k=1,\ldots,K$
M-step: for $k=1,\ldots,K$
            M1.  $\alpha_k^{m+1} \leftarrow \#\mathrm{P}_k^m / N$
            M2.  Compute the mutual information
                 $\hat{I}_k(x_i;x_j)$ between each pair of
                 attributes $i \neq j$.
            M3.  Apply MWST using $\hat{I}_k(x_i;x_j)$ as branch
                 weights between attributes.
            M4.  Transform the undirected tree to a
                 directed one to obtain $P_k^{m+1}$.
```

Figure 4.8 Unsupervised training of CL trees

```
CEM-TAN
Starting from an initial partition $P^0$, the $m^{th}$ iteration of the CEM-TAN:

E-step:     compute $t_k^m(\mathbf{x}^r)$ for $r=1,\ldots,N$ and $k=1,\ldots,K$
S/C-step:   if $m$ < user defined threshold
                for $r=1,\ldots,N$ assign at random each $\mathbf{x}^r$
                to one of the clusters $P_1,\ldots,P_K$ with
                probability $t_k^m(\mathbf{x}^r)$, $k=1,\ldots,K$
            else
                for $r=1,\ldots,N$ assign each $\mathbf{x}^r$ to the
                cluster which provides the maximum
                posterior probability $t_k^m(\mathbf{x}^r)$, $k=1,\ldots,K$
M-step:
            M1. Compute the conditional mutual
                information $\hat{I}(x_i;x_j\,|\,c)$ between each
                pair of attributes $i \neq j$.
            M2. Apply MWST using $\hat{I}(x_i;x_j\,|\,c)$ as branch
                weights between attributes.
            M3. $\alpha_k^{m+1} \leftarrow \#P_k^m / N$ for $k=1,\ldots,K$
            M4. Transform the undirected tree to a
                directed one, then for $k=1,\ldots,K$
                obtain $P_k^{m+1}$.
```

Figure 4.9 Unsupervised training of TAN

```
CEM-SBN
Starting from an initial partition $P^0$, the $m^{th}$
iteration of the CEM-SBN:

E-step:    compute $t_k^m(\mathbf{x}^r)$ for $r=1,...,N$ and $k=1,...,K$
S/C-step:  if $m$ < user defined threshold
               for $r=1,...,N$ assign at random each $\mathbf{x}^r$
               to one of the clusters $P_1,...,P_K$ with
               probability $t_k^m(\mathbf{x}^r)$, $k=1,...,K$
           else
               for $r=1,...,N$ assign each $\mathbf{x}^r$ to the
               cluster which provides the maximum
               posterior probability $t_k^m(\mathbf{x}^r)$, $k=1,...,K$
M-step:
           M1. Compute the conditional mutual
               information $\hat{I}(x_i;x_j \mid c)$ between each
               pair of attributes $i \neq j$.
           M2. Apply MWST using $\hat{I}(x_i;x_j \mid c)$ as branch
               weights between attributes. In each
               iteration:
               M2.1 compute the Bayesian measure $\lambda$
               M2.2 check the $\Lambda$ condition
           M3. $\alpha_k^{m-1} \leftarrow \#P_k^m / N$ for $k=1,...,K$
           M4. Transform the undirected network to
               a directed one, then for $k=1,...,K$
               obtain $P_k^{m+1}$.
```

Figure 4.10 Unsupervised training of SBN

4.5 Methods

To test the three Bayesian network classifiers using the unsupervised training approach, described in the previous section, 10 benchmark datasets from the UCI machine learning repository (Murphy and Aha) were used as well as a data set from a real industrial application: wood defect classification (Estévez *et al.*, 2003).

4.5.1 Benchmark data sets

A brief description of the 10 datasets used in this study appears in Table 4.1. Because Bayesian networks are learned using discrete random variables, continuous attributes were discretized, using a simple unsupervised technique called *Equal Width Interval Binning* (Dougherty *et al.*, 1995), into 5 equally sized bins. As mentioned before, the CEM algorithm can converge to a local optimum which sometimes is far from the optimum solution. In order to reduce this problem, the algorithm starts of by using the S-step for 200 iterations. Then, the best solution (maximum value) according to the CML function is selected as the starting point for the CEM algorithm using the C-step, and this second stage usually converges in no more than 10 iterations. In addition to what was described before, a multistart strategy was also used to reduce convergence to local optima solutions. This strategy consists in starting from ρ different initial points (partition P^0) and then selecting the model that finally scores the highest CML value. In the experiments conducted in this work $\rho = 10$ was used. The performance of the models is evaluated by computing the correct clustering percentage, and this is carried out by voting using the original class label information. For each data set, the performance is obtained as the mean value of 5 repetitions. These results will be compared to the standard clustering techniques k-means and EM algorithm (both with 10 repetitions). Also the results obtained by the three Bayesian network classifiers with supervised learning will be included. The CL Trees and TAN results are the ones reported in Friedman *et al.* (1997) with no smoothing parameters, and the results for the data sets wine and zoo, were the ones reported in Gurwicz and Lerner (2006). The three Bayesian networks models, trained in a supervised way, use 5-fold cross validation.

Table 4.1 Description of the benchmark data sets

Dataset	#Attributes	#Clusters	#Examples
Corral	6	2	128
Crx	15	2	653
Diabetes	8	2	768
Flare	10	2	1066
Glass	9	6	214
Iris	4	3	150
Lymphography	18	4	148
Vote	16	2	435
Wine	13	3	178
Zoo	16	7	101

4.5.2 Wood defect classification

This data was generated from a low-cost Automatic Visual Inspection (AVI) system for wood defect detection (Estévez *et al.*, 2003). The AVI systems usually include the following stages (Pham and Alcock, 2003): 1) *Image acquisition*: to obtain an image of the object to be inspected; 2) *Image enhancement*: to improve the quality of the acquired image, which facilitates later processing; 3) *Image segmentation*: to divide the image into areas of interest and background. The result of this stage is called the segmented image, where *objects* represent the areas of interest; 4) *Feature extraction*: to calculate the values of parameters that describes each object; 5) *Classification*: to determine what is represented by each object.

The feature extraction module from the AVI system in Estévez *et al.* (2003) extracted features from objects and windows of 64 by 64 pixels centred in the object geometrical centre. The features used in this work include: 7 object geometrical features measured on the binarised grey image (e.g.: area, perimeter, average radius, aspect ratio, etc.); 96 object colour features (24 features measured in each of the four channels, R, G, B, and grey); 46 window colour features (e.g., mean and variance of window histograms, mean and variance at the edge of windows); and 16 co-occurrence features (which contain texture information) which were added recently in Ruz *et al.* (2007). In total there are 165 features computed from the segmented defects. The data set used in this work consists in 2247 examples (wood board images of 320 × 240 pixels) which have been manually labelled/classified into one of the following 10 defect categories (see Ruz *et al.* (2007) and Ruz *et al.* (2005) for details): birdseye, pockets, wane, split, stain, blue stain, pith, dead knot, live knot, and hole. Also, the clear wood category was added, which corresponds to non-defective wood board images. In total there are 11 classes, with approximately 200 examples per class.

For this data set, the CL multinet clustering performance was analysed and compared to the k-means and the EM algorithm. Since this data set contains a significantly larger amount of attributes compared to the standard benchmark dataset described before, to speed up the learning process for the CL multinet, the results presented are the average value of five repetitions with $\rho = 5$ and 50 iterations for the S-step and 5 iterations for the C-step.

4.5.3 Initialisation

In order to obtain the initial partition P^0, K points are selected randomly from the data set, where K is the number of clusters that will be formed. Then, by computing the Euclidean distance between each data point from the data set with each one of the K points, assign the data point to the partition (one of the K points) that has the smallest distance.

Another common practice when using the EM framework is to initialise using the k-means (Roberts *et al.*, 2000; Ueda and Nakano, 1998) , but since in this work the performance of the Bayesian network classifiers will be compared to the k-means, only the random start was considered.

4.6 Experimental results and discussions

The results using the benchmark datasets are shown in Table 4.2. In general the three Bayesian network classifiers models out-perform the traditional clustering algorithms, except for the wine data set where the EM algorithm performs better than the three models and k-means is better than the CEM-CL trees and similar to CEM-TAN. This particular case could be due to the simple discretization method used for the continuous attributes in the benchmark data sets, and it is possible that more bins are needed for the wine data set, whereas the k-means and EM use the original continuous attributes. Finding the optimum number of bins to use in the unsupervised discretization method for each data set was not considered, since in general for the ten data sets tested, a discretization into 5 bins obtained good results.

For the Flare, Iris, and Vote data sets, the unsupervised training of the three Bayesian networks classifiers obtained similar results to the three models trained in a supervised way (last three columns of Table 4.2). Data sets Crx, Diabetes, Lymphography, Wine, and Zoo have less than 10% difference with the supervised results, whereas Corral and Glass perform significantly worse than the supervised results but much better than the traditional clustering algorithms, specially k-means. The EM algorithm for data sets Diabetes, Flare, Iris, Vote, and Wine converged always to the same sub-optimal solution in the 10 repetitions, and although the results obtained by the Bayesian network models are sub-optimal as well, convergence to the same local optima is avoided thanks to the S-step. Another way to avoid this problem is to incorporate a simulated annealing approach described in Celeux and Govaert (1992) and Ueda and Nakano (1998). The effects of the simple Bayesian network classifier SBN, which automatically regulates the number on edges (complexity of the network), can be appreciated by analysing the first data set in Table 4.2. The Corral data set contains 6 attributes and the class variable is a Boolean function of only four of the attributes: $(1 \wedge 2) \vee (3 \wedge 4)$. The fifth attribute is entirely irrelevant and the sixth attribute is "correlated" with the class variable in that it matches the class label 75% of the time. The structures for the best results obtained out of the 5 repetitions for the unsupervised training of the CL trees, TAN, and SBN are shown in Fig. 4.11, Fig. 4.12, and Fig. 4.13. Note that the edges from C to the attributes are dotted because in clustering the C variable is latent. From Fig. 4.13 it is clear that attribute 5 does not depend probabilistically on the other attributes, given C, and the joint probability distribution (jpd) using Eq. (2.2) is

$$P(C)P(1\,|\,C)P(5\,|\,C)P(2\,|\,1,C)P(3\,|\,1,C)P(4\,|\,3,C)P(6\,|\,3,C)$$

(once the model is trained and the C variable contains cluster labels). Because attribute 5 is irrelevant, in fact, $P(5\,|\,C=0)=0.5$ and $P(5\,|\,C=1)=0.5$ for any value that attribute 5 takes, $P(5\,|\,C)$ has no effect in the jpd computation which is what it is expected. On the other hand, the TAN model is restricted to building tree structures, and in this case, attribute 5 participates in the jpd by $P(5\,|\,1,C)$. This conditional probability also remains constant and equal to 0.5 regardless of the values that 5, 1, and C take, but requires unnecessary extra computation compared to $P(5\,|\,C)$. Also from Fig. 4.12 it is not clear that attribute 5 is "different" from the rest of the attributes. It has the same conditioning (parents) as attributes 3, 2, and 6 whereas in Fig. 4.13, it is clear that attribute 5 is different from the other attributes. Also from Table 4.2, the performance of the SBN, both in unsupervised and supervised learning, in most of the cases is superior to the TAN model, thanks to its more realistic representation of the data, which is not restricted to a tree structure all the time.

The performance of the CL multinets on the wood data set appears in Table 4.3, and although the computational cost is high for the 165-dimensional input space considering that each tree learning procedure in the multinet has time complexity $O(n^2 n_k)$, the results obtained are considerably better than the traditional algorithms, especially when compared to the k-means. In these cases where there are a large number of attributes, the CL multinets as well as the other two Bayesian network models can be initialised using k-means in order to reduce the learning time. To prove this, a simple experiment was carried out using the best solution of the k-means (55.09%) as the starting partition P^0, the CL multinet reached a performance of 65.42 % in only 15 iterations (10 in the S-step and 5 in the C-step).

One of the limitations of the proposed unsupervised training method for the three Bayesian network models is that the number of clusters must be known a priori. This drawback is common for several clustering algorithms (k-means, EM, etc.) but can usually be overcome by learning models with different numbers of clusters and then selecting the best model according to some cluster validity index (Bezdek and Pal, 1998).

Table 4.2 Experimental results using the benchmark data sets

Dataset	CEM-CL Trees	CEM-TAN	CEM-SBN	K-Means	EM	CL Tress	TAN	SBN
Corral	83.12±2.79	77.50±3.42	80.00±2.79	67.81±9.43	74.21±1.98	99.23±0.77	95.32±2.26	96.80±3.35
Crx	79.57±2.32	78.19±3.53	80.82±2.13	77.04±3.78	74.42±11.09	83.92±1.05	83.77±1.34	87.54±1.74
Diabetes	70.44±3.38	70.80±2.97	71.38±2.13	66.79±0.00	66.01±0.00	74.35±1.43	75.13±0.98	78.30±3.18
Flare	81.57±1.56	80.35±0.75	81.03±1.04	64.42±8.98	79.08±0.00	81.90±1.51	82.74±1.60	83.38±1.27
Glass	48.50±2.17	52.33±3.32	51.68±3.23	43.73±3.52	44.71±1.37	69.17±1.29	69.18±2.64	70.00±2.71
Iris	90.66±3.68	92.00±2.30	92.40±2.43	88.26±0.34	90.66±0.00	93.33±1.05	93.33±1.05	96.00±3.65
Lymphography	62.97±4.83	62.83±5.12	62.16±5.50	44.32±8.19	48.98±5.71	64.11±4.77	66.87±3.37	68.28±4.50
Vote	89.28±1.56	88.18±1.57	88.00±0.86	86.71±0.14	87.81±0.00	89.42±1.72	89.20±1.61	90.34±2.09
Wine	93.37±3.58	94.83±2.07	95.05±1.08	94.66±0.29	97.19±0.00	98.27±1.65	98.03±1.55	97.14±2.02
Zoo	86.93±1.90	87.72±2.58	88.71±2.27	83.16±2.64	83.26±3.66	93.09±5.03	95.08±4.25	94.00±4.18

Table 4.3 Experimental results using the wood data set

Dataset	CEM-CL Trees	K-Means	EM
Wood data	65.60±1.30	52.78±3.57	57.74±1.81

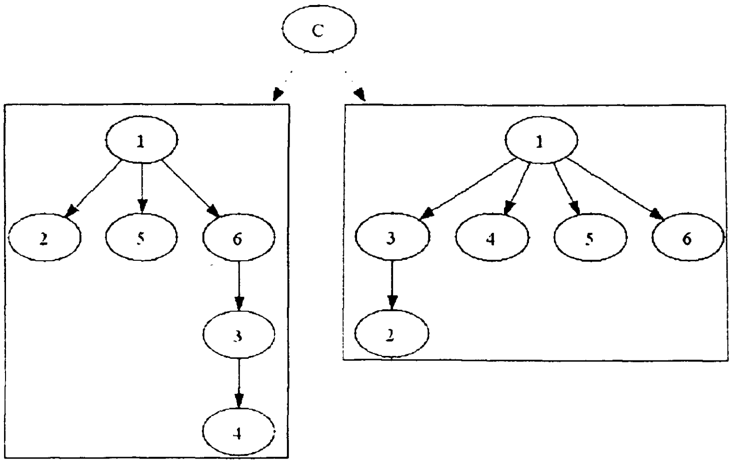

Figure 4.11 CL trees of the "Corral" data set

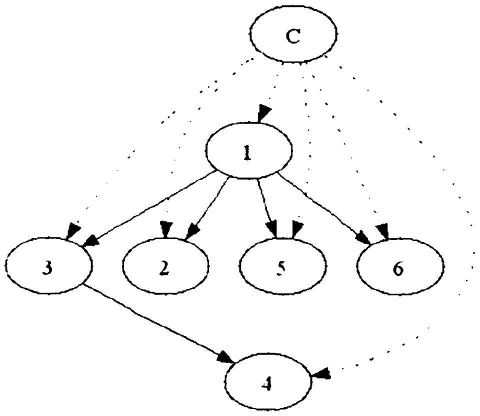

Figure 4.12 TAN of the "Corral" data set

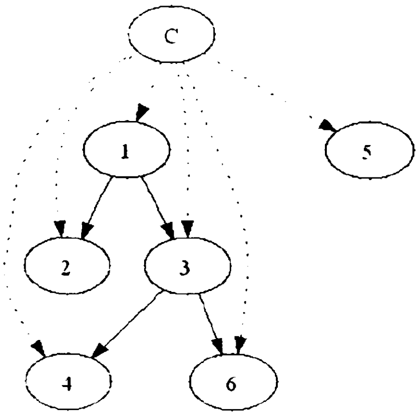

Figure 4.13 SBN of the "Corral" data set

4.7 Summary

Clustering using Bayesian network classifiers has been addressed in the present chapter. To accomplish the unsupervised training of the Bayesian network classifiers, which includes structure and parameter learning, as well as the clustering of the data, the maximisation of the classification maximum likelihood was used. This maximisation was carried out by formulating the three Bayesian network classifiers: CL multinets, TAN, and the SBN classifier (introduced in chapter 3) into the Classification EM algorithm framework. Results using benchmark data sets as well as real-world applications show that Bayesian network classifiers are promising models for clustering tasks. The SBN classifier generally out performs the TAN model when trained in a supervised or unsupervised way. This is due to its more robust structure learning process, which does not limit its structure to a tree and regulates the number of edges according to the amount of training data available which determines the level of complexity in the network structure.

For future work, two of the main limitations of the proposed method should be researched. These are: the difficulty of determining the correct number of clusters during the learning process, and the need for a more advanced unsupervised technique for transforming continuous attributes into discrete ones.

Chapter 5

TRAINING NAIVE BAYES MODELS FOR DATA VISUALISATION

5.1 Preliminaries

The naive Bayes model is considered one of the top 10 data mining algorithms due to its simplicity, robustness, elegance, and effectiveness (Wu *et al.*, 2008). The models' main characteristic is its feature independence assumption. The surprisingly good performance of the of the naive Bayes model for classification has been studied in Domingos and Pazzani (1997) and Friedman (1997). In Domingos and Pazzani (1997) they found that the naive Bayes classifier performs quite well in practice even when strong feature dependences are present, they show that this follows partly because, contrary to previous assumptions, the naive Bayes classifier does not require feature independence to be optimal under zero-one loss (misclassification rate). On the other hand, Friedman (1997) analysed how the bias and variance components of the estimation error combine to influence the classification performance. They show, under certain conditions, that the low variance associated with the naive Bayes model can significantly reduce the effect of the high bias due to the strong feature independence assumptions.

In cases where the model obtains poor results, a straightforward solution is to perform a feature selection stage before using the naive Bayes classifier. An example of this procedure can be found in the Selective Bayesian Classifier (Langley and Sage, 1994). High dimensional data projected on to a lower dimension (usually 2D) enables human beings to explore the natural structure of the data and commonly used in data mining to visually identify clusters in the data set.

Amongst the many projections/visualisation techniques, principal component analysis (PCA), the Self Organising Maps (SOM) (Kohonen, 1982; Kohonen, 1995), and the Generative Topographic Mapping (GTM) (Bishop *et al.*, 1998) are of great interest, since most of the ongoing research and new developments are based on one of these techniques.

The PCA performs linear projections by using the first and second principal eigenvectors, computed from the data covariance matrix. Kohonen's SOM performs a nonlinear projection from a high dimensional data space onto a 2D neuron grid. Each neuron in the output space has associated a weight vector in the input space. The training consists in moving the weight vectors, given a new stimulus, in order to model the input distribution. The updates of the weights are carried out by the SOM rule, then once the training process has finished, each data point is assigned to the closest weight vector yielding topology preserving maps.

The GTM is a probabilistic reformulation of the SOM. The output is characterised by a 2D latent variable space. The idea behind the GTM is to project the latent variables onto the input space through a radial basis function (RBF) network. The projected latent variables in the input space define the centres of radially symmetric gaussians. The input distribution is then modelled by a mixture of these gaussians and the training process is carried out by the EM algorithm. Once the models' parameters have been learned (RBF network parameters and the radially symmetric gaussians variance) the inverse projection from the input space to the latent space is done by the Bayes' rule.

Apart from these well know projection methods, the work of Van Hulle (2002), and Yin and Allinson (2001) are also important for this chapter. In Van Hulle (2002) a new learning algorithm for kernel-based topographic map formation is presented. The learning algorithm procedure aims at maximising the joint entropy of the map's output while in Yin and Allinson (2001) a self-organising mixture network is derived for learning arbitrary density functions. The learning procedure is based on minimising the Kullback-Leibler divergence using stochastic approximation and a self organising principle.

Because of the great popularity that the naive Bayes models have in classification and clustering tasks, it is natural to think of exploring some other fields where this simple model can perform just as well.

This chapter presents a new unsupervised training method for the naive Bayes models in order to generate topology preserving maps. The paper is organized as follows: section 5.2 presents a brief description of naive Bayes models for clustering. The proposed naive Bayes mapping technique is developed in section 5.3. Simulations on several examples and comparisons with well known mapping techniques are carried out in section 5.4. Section 5.5 comments on the relation of the proposed algorithm with other existing methods. The final conclusions are presented in section 5.6.

5.2 Naive Bayes models

Naive Bayes models have been widely used for classification and clustering tasks, for this present work, the latter will be of interest. Given an observation of an n-dimensional vector variable $\mathbf{x} = (x_1, \ldots, x_n)$ (each variable x_i can be continuous or discrete), the joint probability distribution over $\mathbf{x}$ can be computed using a naive Bayes model for clustering, defined as

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \prod_{i=1}^{n} p(x_i \mid c = k) \tag{5.1}$$

$$\sum_{k=1}^{K} \pi_k = 1, \qquad \pi_k \geq 0$$

where c is a discrete hidden variable "cluster" which causes the variables to be (conditionally) mutually independent. The values of c, $k = 1, \ldots, K$, indicates the different clusters in the data and π_k the probability distribution of the variable "cluster". This model (5.1) can also be interpreted as a mixture model of K components with mixing weights π_k and mixing component distributions $p(\mathbf{x}|k) = \prod_{i=1}^{n} p(x_i \mid k)$.

The naive Bayes model has a Bayesian network representation which is shown in Fig. 5.1. Training of this clustering model can be carried out by adopting the *maximum likelihood* approach via the EM algorithm (Dempster *et al.*, 1977). After initialising the model's parameters (mixing weights and the parameters of each component distribution) , the training consists in alternating between the E-step in which the posterior probability of each data point belonging to one of the K clusters is computed, and the M-step in which the parameters of the model are updated in order to maximise the log likelihood. For more details of the naive Bayes model for clustering purposes refer to (Chickering and Heckerman, 1997; Meilă and Heckerman, 1998).

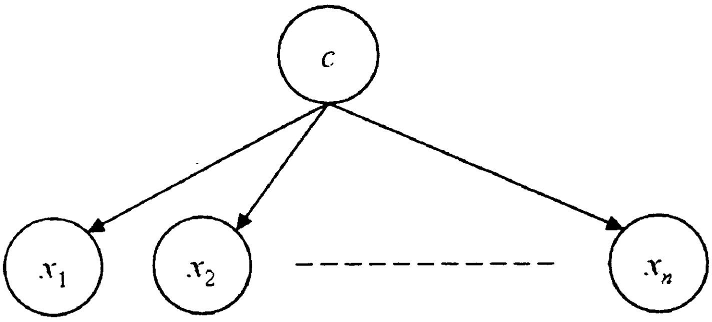

Figure 5.1 Naive Bayes model

5.3 Training of naive Bayes models for topographic map formation

The previous section presented a brief description of a naive Bayes model for clustering, in order for this model to generate topographic maps (e.g. SOM) additional definitions and modifications are introduced. First, a discrete neuron grid with K neurons is defined as the out-put space Y. Second, each cluster label $c = k$ with $k = 1,\ldots,K$ has associated a coordinate position in the out-put space $\mathbf{y}_k = (y_{1k}, y_{2k})$. Third, each neuron (grid position $\mathbf{y}_k$) has equal prior probability, i.e. $\pi_k = 1/K$ for all k. Fourth, each variable of the input data x_i follows a normal density distribution for a given neuron k

$$p(x_i \mid k) = \frac{1}{(2\pi\sigma_{ik}^2)^{\frac{1}{2}}} \exp\left(-\frac{(x_i - \omega_{ik})^2}{2\sigma_{ik}^2}\right), \qquad (5.2)$$

where ω_{ik} and σ_{ik}^2 are the mean and variance respectively, for the *ith* variable and the *kth* neuron.

Using the previous formulation, the distribution of $\mathbf{x}$, for a given c=k and $\boldsymbol{\theta}_k = \{\boldsymbol{\omega}_k, \boldsymbol{\sigma}_k^2\}$ with $\boldsymbol{\omega}_k = \{\omega_{1k},\ldots,\omega_{nk}\}$, and $\boldsymbol{\sigma}_k^2 = \{\sigma_{1k}^2,\ldots,\sigma_{nk}^2\}$, is defined by

$$p(\mathbf{x} \mid k, \boldsymbol{\theta}_k) = \prod_{i=1}^{n} p(x_i \mid k), \qquad (5.3)$$

and the distribution of $\mathbf{x}$ in the input data space is given by

$$p(\mathbf{x} \mid \boldsymbol{\Theta}) = \frac{1}{K}\sum_{k=1}^{K} p(\mathbf{x} \mid k, \boldsymbol{\theta}_k), \qquad (5.4)$$

with $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_K\}$.

The aim of the training process is to determine the model's parameters $\mathbf{\Theta}$ given a data set $D = (\mathbf{x}^1, \ldots, \mathbf{x}^N)$ with N vector points which can be carried out by using maximum likelihood. For practical reasons, instead of maximising the likelihood, it is convenient to use the log-likelihood, defined by

$$LL(\mathbf{\Theta}) = \log \prod_{j=1}^{N} p(\mathbf{x}^j \mid \mathbf{\Theta}). \tag{5.5}$$

For the specific joint probability distribution in (5.4), the log-likelihood function can be expresses as

$$LL(\mathbf{\Theta}) = \sum_{j=1}^{N} \log\left(\frac{1}{K} \sum_{k=1}^{K} p(\mathbf{x}^j \mid k, \mathbf{\theta}_k) \right). \tag{5.6}$$

To maximise this function several optimisations techniques could be used but because the distribution (5.4) follows a mixture model, it is preferable to use the EM algorithm. In order to handle large data sets, Titterington's online version of the EM algorithm (Titterington, 1984) will be used. Titterington showed that for a mixture of univariate Normals, with mean ω_k and variance σ_k^2 and mixing weight π_k, the updating rules for the $(j+1)th$ input (iteration) data are:

E-step

$$\gamma_k^{(j)} = p(k \mid x^{(j+1)}, \omega_k^{(j)}, \sigma_k^{2(j)}) = \frac{\pi_k^{(j)} p(x^{(j+1)} \mid k, \omega_k^{(j)}, \sigma_k^{2(j)})}{\sum_{k'=1}^{K} \pi_{k'}^{(j)} p(x^{(j+1)} \mid k', \omega_{k'}^{(j)}, \sigma_{k'}^{2(j)})}. \tag{5.7}$$

M-Step

$$\pi_k^{(j+1)} = \pi_k^{(j)} + j^{-1}(\gamma_k^{(j)} - \pi_k^{(j)}), \tag{5.8}$$

$$\omega_k^{(j+1)} = \omega_k^{(j)} + f_k^{(j)}(x^{(j+1)} - \omega_k^{(j)}), \tag{5.9}$$

$$\sigma_k^{2(j+1)} = \sigma_k^{2(j)} + f_k^{(j)}\left((x^{(j+1)} - \omega_k^{(j)})^2 - (1 - f_k^{(j)})\sigma_k^{2(j)}\right), \tag{5.10}$$

where

$$f_k^{(j)} = \frac{\gamma_k^{(j)}}{j\pi_k^{(j)} + \gamma_k^{(j)}}. \tag{5.11}$$

The idea behind these updating rules is to distribute the effect of the new observation to all the terms in proportion to their respective likelihoods. The mixing weight, mean, and variance are then updated by this proportion. Convergence results for this online EM algorithm have been given by Titterington (1984).

Note that the naive Bayes model, for any k, consists of the product of n univariate Normals. So, because of the models' attribute independence assumption, the same updating rules can be used for each dimension.

So far, no topology preserving restrictions have been made and the above set of updating learning rules will not yield topographic maps because each mixing distribution centre can move in a free manner in the input space. To achieve topology preserving lattices in the learning process, a posterior probability-based competition between the neurons is introduced. The "winning" neuron is defined as $k^* = \arg\max_{\forall k \in \{1,\ldots,K\}} \gamma_k$, in other words, for a given input data the winning neuron is the one with the highest posterior probability for that data. Then, topological information is incorporated by means of a neighbourhood function Λ which is a monotonically decreasing function of the distances (in the out-put space) of a neuron k and the winner k^*. The most common function is the Gaussian function, which will be used in this work, defined by

$$\Lambda(k,k^*,\sigma(t)) = \exp\left(-\frac{\left\|\mathbf{y}_k - \mathbf{y}_{k^*}\right\|^2}{2\sigma^2(t)} \right), \tag{5.12}$$

where $\sigma(t)$ is the neighbourhood function range.

Now, returning to the updating rules, by incorporating the W-step (winning neuron) after the E-step and before the M-step, and in the M-step taking $\gamma_k = \delta_{kk^*}$ and introducing the neighbourhood function, the updating rules that yield topology-preserving maps for naive Bayes models are:

E-step

$$\gamma_k^{(j)} = p(k \mid \mathbf{x}^{(j+1)}, \boldsymbol{\omega}_k^{(j)}, \boldsymbol{\sigma}_k^{2(j)}) = \frac{p(\mathbf{x}^{(j+1)} \mid k, \boldsymbol{\omega}_k^{(j)}, \boldsymbol{\sigma}_k^{2(j)})}{\sum_{k'=1}^{K} p(\mathbf{x}^{(j+1)} \mid k', \boldsymbol{\omega}_{k'}^{(j)}, \boldsymbol{\sigma}_{k'}^{2(j)})}. \tag{5.13}$$

W-step

$$k^* = \arg\max_{\forall k \in \{1,\ldots,K\}} \gamma_k^{(j)}. \tag{5.14}$$

M-step

$$\boldsymbol{\omega}_k^{(j+1)} = \boldsymbol{\omega}_k^{(j)} + f_k^{(j)}(\mathbf{x}^{(j+1)} - \boldsymbol{\omega}_k^{(j)}), \tag{5.15}$$

$$\boldsymbol{\sigma}_k^{2(j+1)} = \boldsymbol{\sigma}_k^{2(j)} + f_k^{(j)}\left((\mathbf{x}^{(j+1)} - \boldsymbol{\omega}_k^{(j)})^2 - (1 - f_k^{(j)})\boldsymbol{\sigma}_k^{2(j)}\right), \tag{5.16}$$

where

$$f_k^{(j)} = \frac{\Lambda(k,k^*,\sigma(t))}{j(1/K) + \Lambda(k,k^*,\sigma(t))}. \tag{5.17}$$

By introducing the neighbourhood function, maximisation of the log-likelihood function is still performed but in a constrained manner that helps the developments of topology preserving maps.

Once the training has finished, visualisation can be performed by computing, for each data point, its posterior probability (5.13). Then, a *hard* projection or a *soft* projection can be made. The hard projection consists in assigning a data point to the coordinate $\mathbf{y}_{k^*}$ with k^* found using (5.14). The soft projection consists in computing the posterior mean projection of a data point *j*, defined by

$$\mathbf{y} = \sum_{k=1}^{K} \gamma_k^j \mathbf{y}_k \,. \qquad (5.18)$$

5.4 Simulations

Several simulations with artificial data and real industrial data are reported in this section, as well as comparisons with well known data visualisation techniques.

5.4.1 Map formation example

The map formation process, in the input space, by iterating equations (5.13) to (5.16) is shown in Fig. 5.2. The map is of size 16 x 16 and the mean vectors and the variance vectors are 2-dimensionl. At the beginning $t = 0$, the mean vectors and the variance vectors are randomly distributed in the unit square. Then for each iteration (updating the rules (5.13) to (5.16)) an input data $\mathbf{x}$ is selected randomly from the unit square. For this simulation, the neighbourhood function range was computed as

$$\sigma(t) = \sigma_0 \exp\left(-\frac{t}{t_{\max}}\right), \qquad (5.19)$$

for $0 \leq t < t_{\max} = 100000$ and σ_0 equal to half of the lineal dimension of the grid. Although any other monotonically decreasing function for the range would be useful as well. Fig. 5.2(a) – Fig. 5.2(f) show results for *t*=0, 500, 1000, 5000, 10000, and 100000 respectively. You can see how the lattice has already been formed by *t*=500 and as $\sigma(t)$ decreases the map stretches out. In the final stages when $\sigma(t)$ is very small no ordering is performed and the algorithm only updates for the winner (competitive learning).

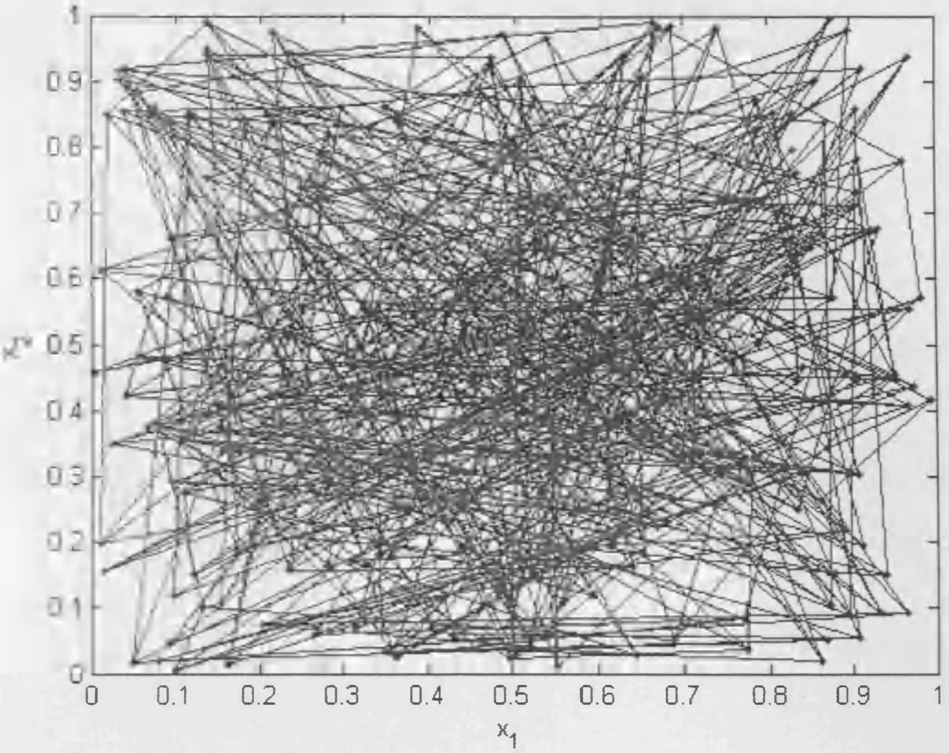

Figure 5.2(a) Naive Bayes map formation at *t* = 0.

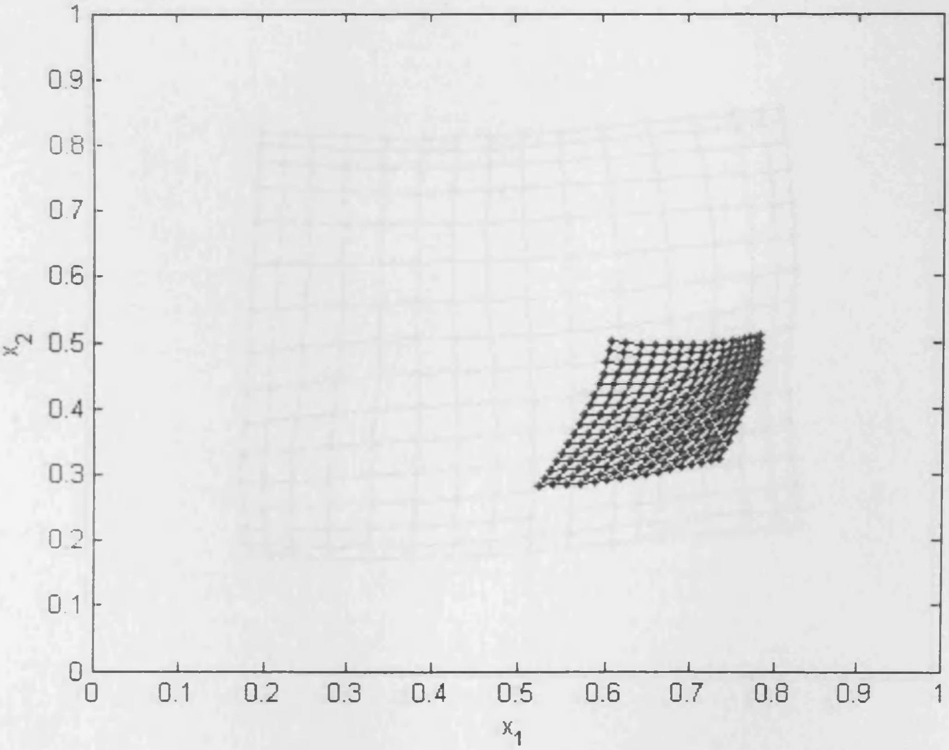

Figure 5.2(b) Naive Bayes map formation at *t* = 500.

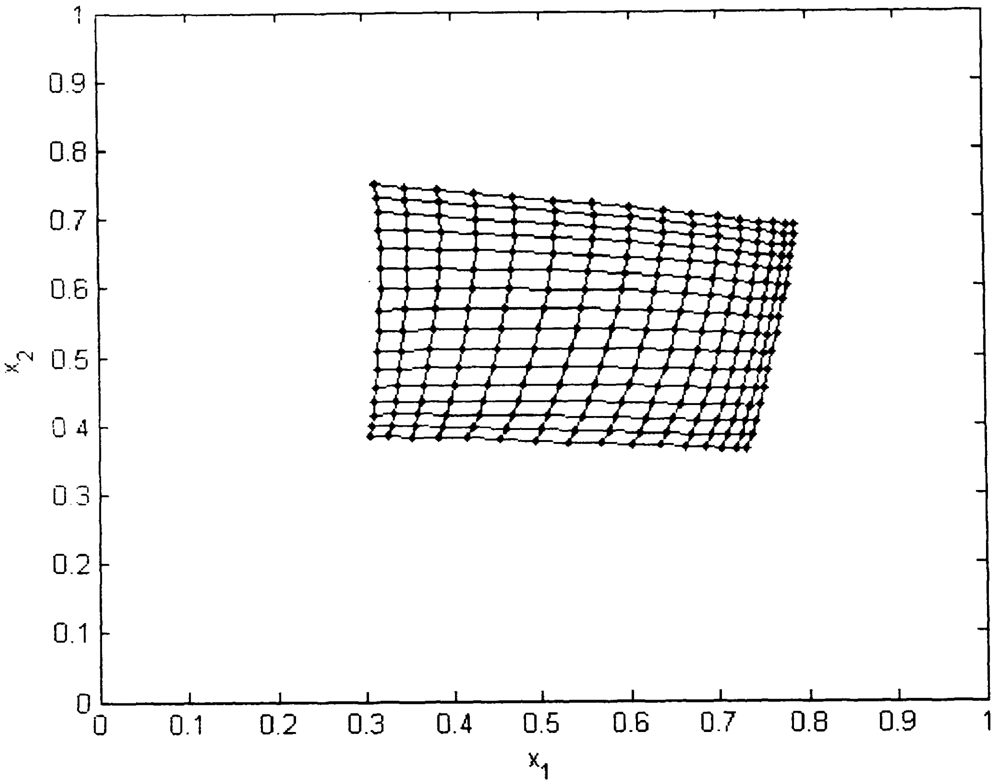

Figure 5.2(c) Naive Bayes map formation at t = 1000.

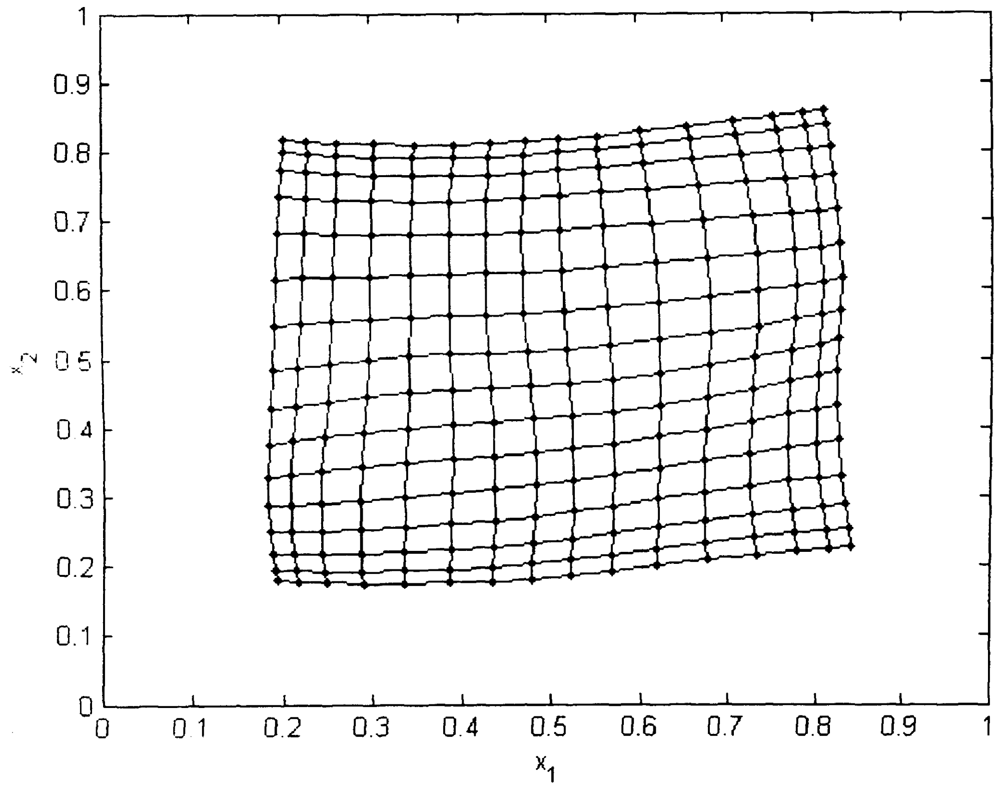

Figure 5.2(d) Naive Bayes map formation at t = 5000.

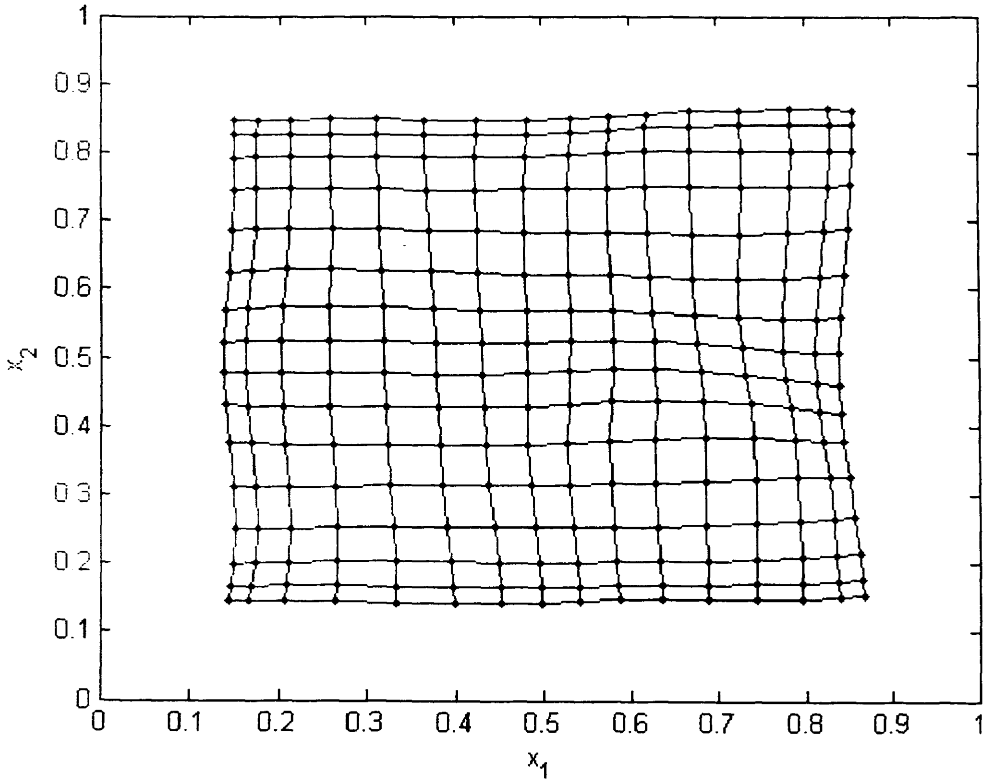

Figure 5.2(e) Naive Bayes map formation at t = 10000.

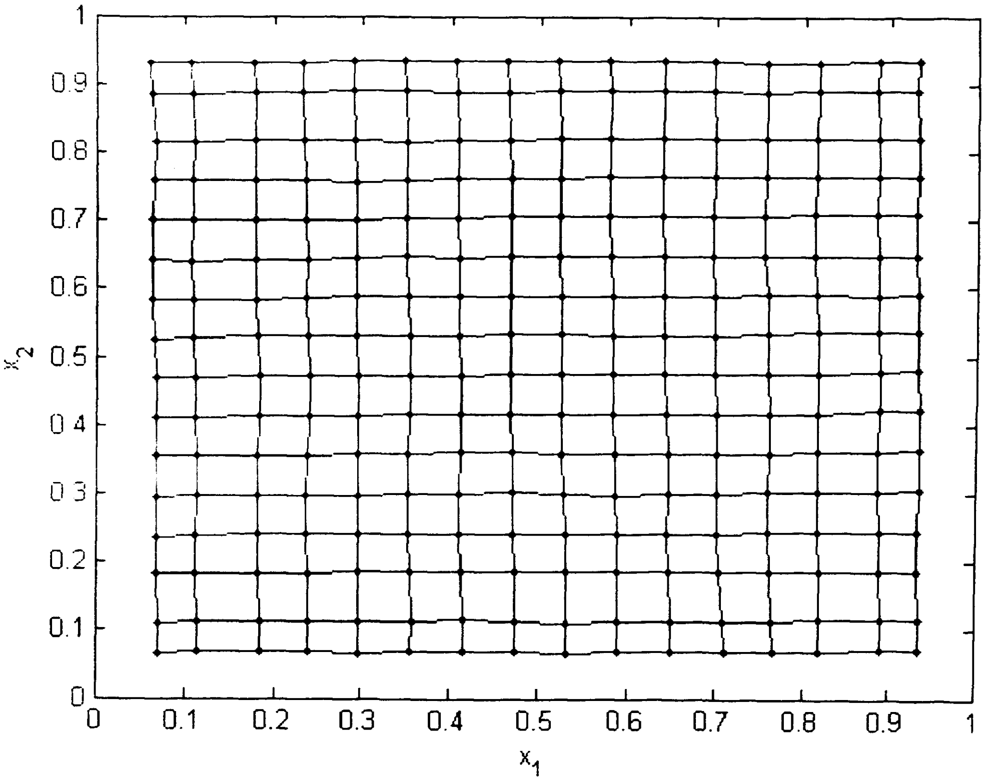

Figure 5.2(f) Naive Bayes map formation at t = 100000.

5.4.2 Monitoring convergence

Because the training of the naive Bayes model for visualisations starts of with the parameters selected randomly several runs can be made starting form different random values and selecting the final model which scores the highest in (5.6) .To show the usefulness of having (5.6) to monitor the convergence in the training phase as well as to see how good the model has fitted the data, an example using the Iris data set (see Table 5.1 for description) from Murphy and Aha is presented. For this simulation, an epochs loop has been introduced, outside the data loop like the SOM algorithm, now the range (5.19) will be reduced for every epoch and the j (iteration number) in (5.17) becomes j + *epoch*N* with $0 \leq epoch < epoch_{\max} = 100$. The results from three examples (outputs) are shown in Fig. 5.3(a), Fig. 5.3(b), and Fig. 5.3(c) respectively and the resulting log-likelihood for each one appears in Fig. 5.3(d). Notice how for the three examples during the first 10 epochs is when most significant improvements in the LL are made, after that, convergence turns slow and only minor improvements are achieved. From Fig. 5.3(d), the example that obtained the best result was number three, and by analysing the output mappings Fig. 5.3(c) appears to be the mapping which best separates de 3 classes, specially between class "+" and "O". In Fig. 5.4(a) – Fig.5.4(c) the evolution of (5.13) for data point number 30 in the Iris data set is presented. Fig. 5.4(a) show the initial posterior probability for that data point which can be interpreted as the membership value of belonging to one of the output neurons. The posterior probability at epoch=10 appears in Fig. 5.4(b) and, in accordance to what Fig. 5.3(d) showed, now membership values are reduced to only a specific region of the map. Finally, at epoch=100, Fig. 5.4(c) shows minor modification to the previous figure and membership values are slightly increased due to a small reduction in the region the was covering before. The final output after applying (5.18) is shown in Fig. 5.4(d) and data point 30 which belongs to the "*" class is represented by a "□".

Table 5.1 Description of the benchmark data sets

Dataset	#Attributes	#Classes	#Examples
Australian	14	2	690
Breast	10	2	683
Cleve	13	2	296
Crx	15	2	653
Diabetes	8	2	768
Heart	13	2	270
Ionosphere	34	2	351
Iris	4	3	150
Pima	8	2	768
Wine	13	3	178

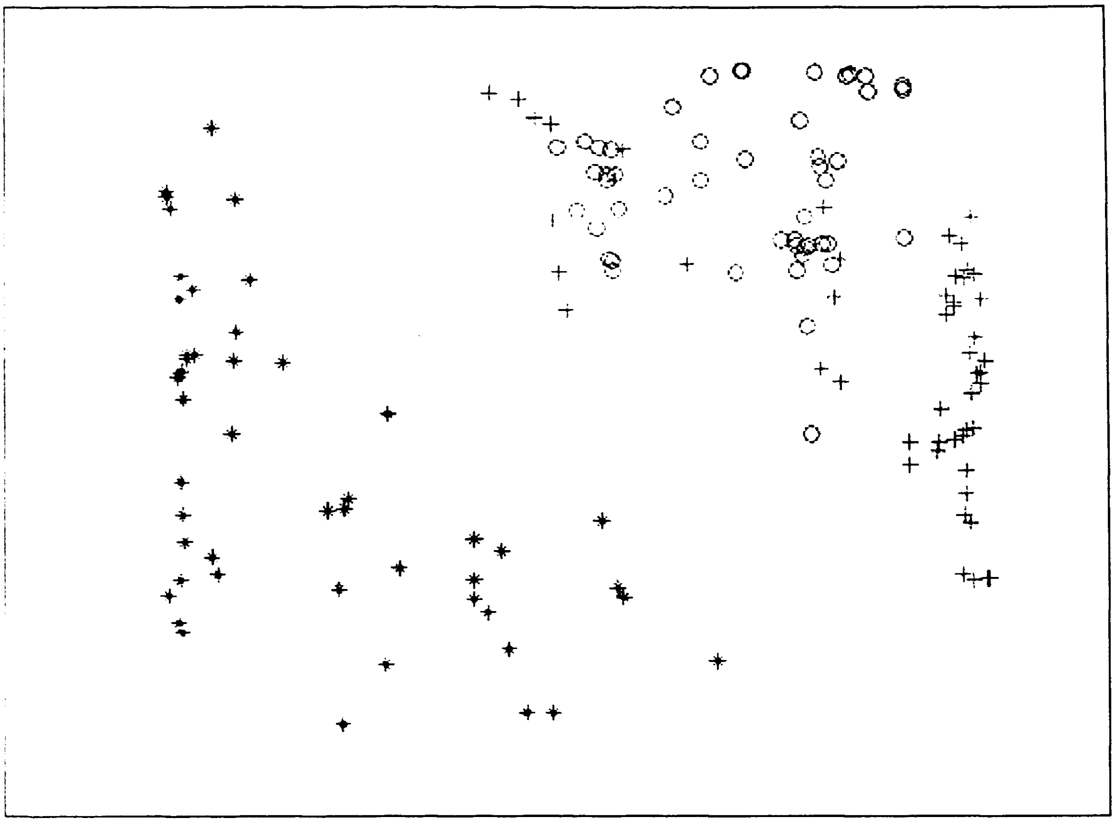

Figure 5.3(a) Output of example 1

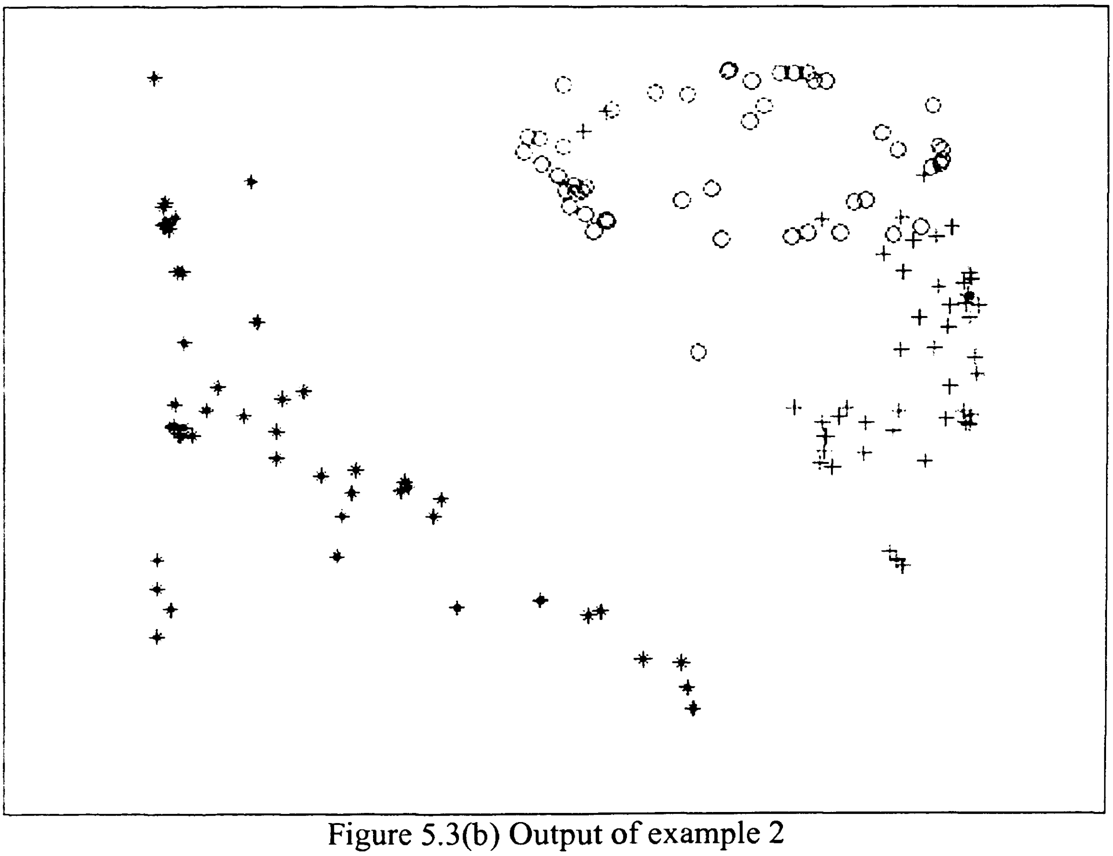

Figure 5.3(b) Output of example 2

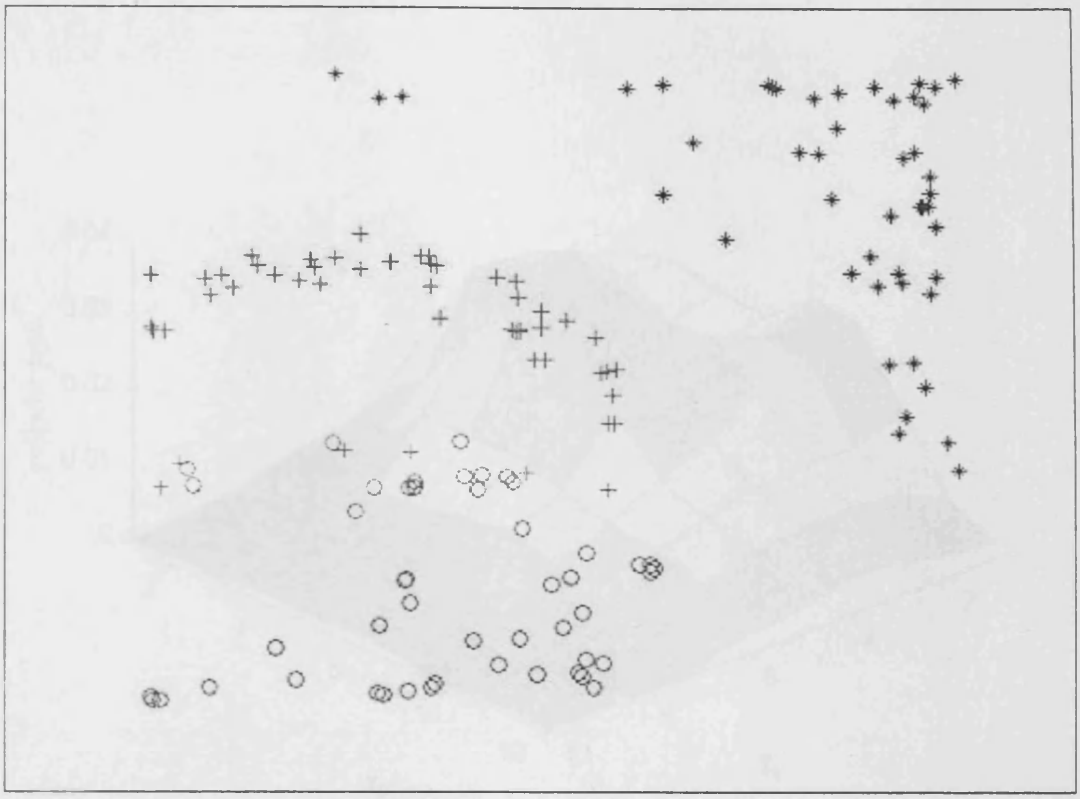

Figure 5.3(c) Output of example 3

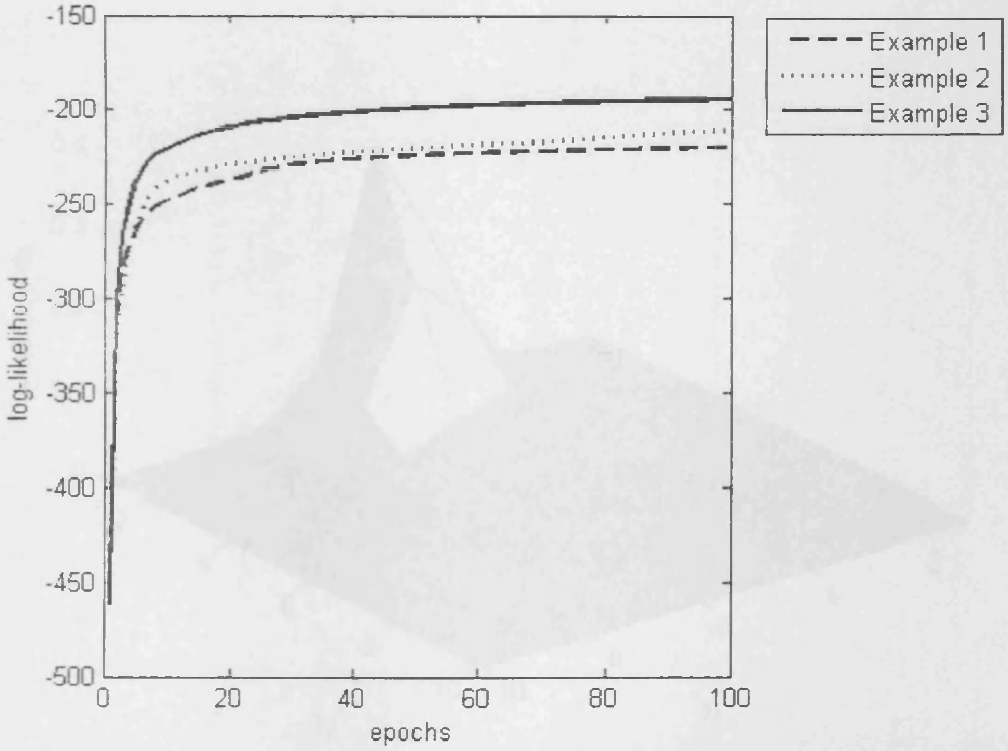

Figure 5.3(d) Log-likelihood convergence for the three examples

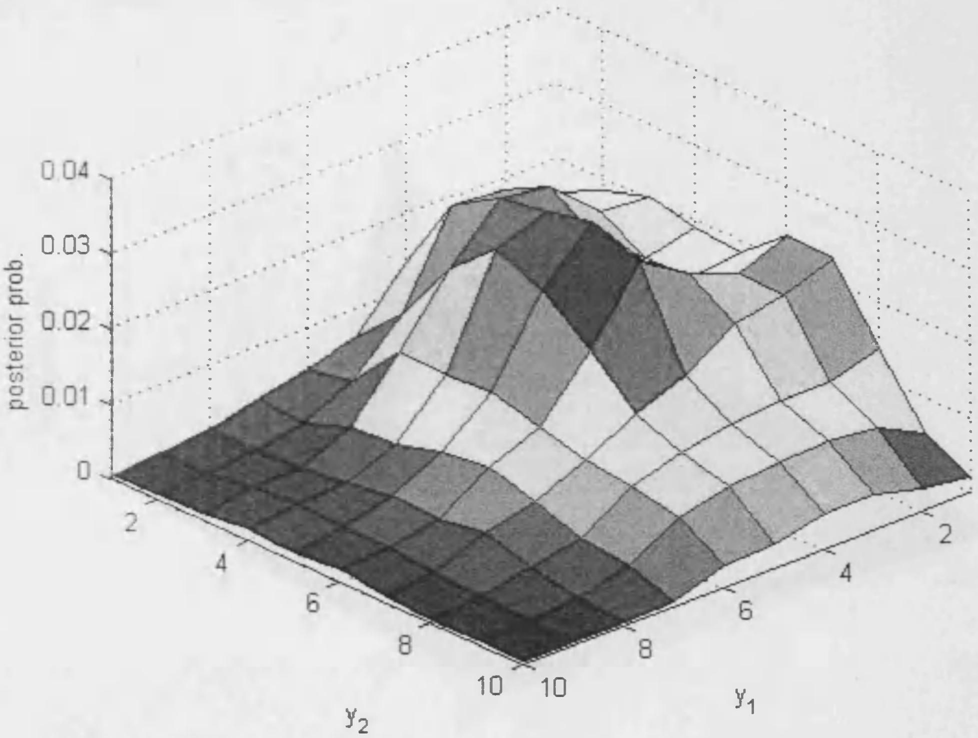

Figure 5.4(a) Initial posterior probability for data point number 30 in the Iris data set

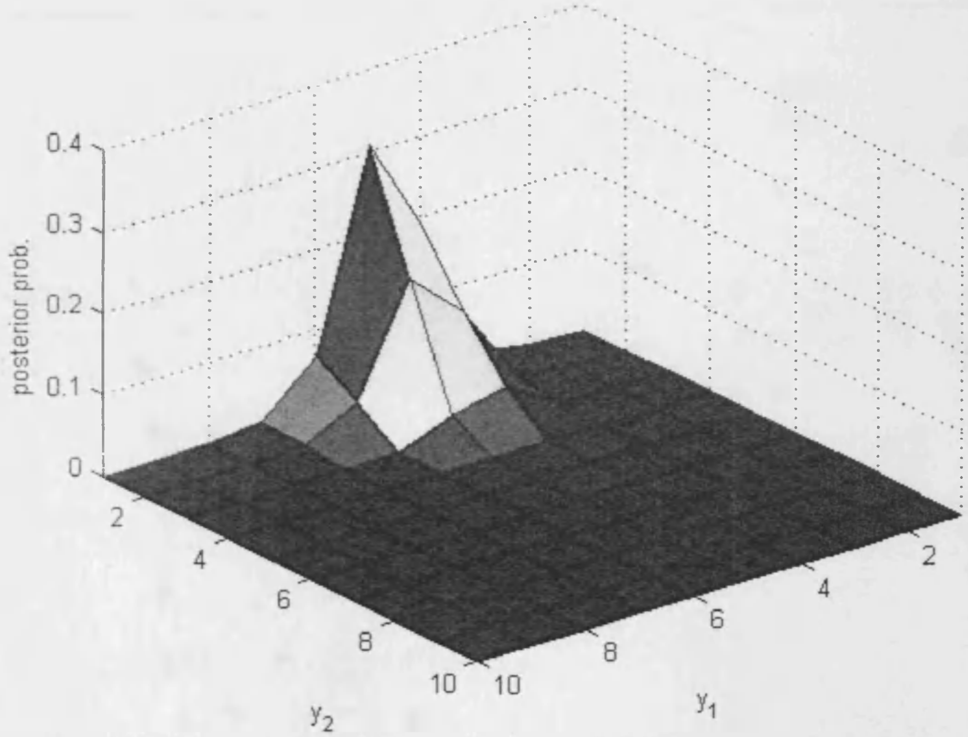

Figure 5.4(b) Posterior probability at epoch=10 for data point number 30 in the Iris data set

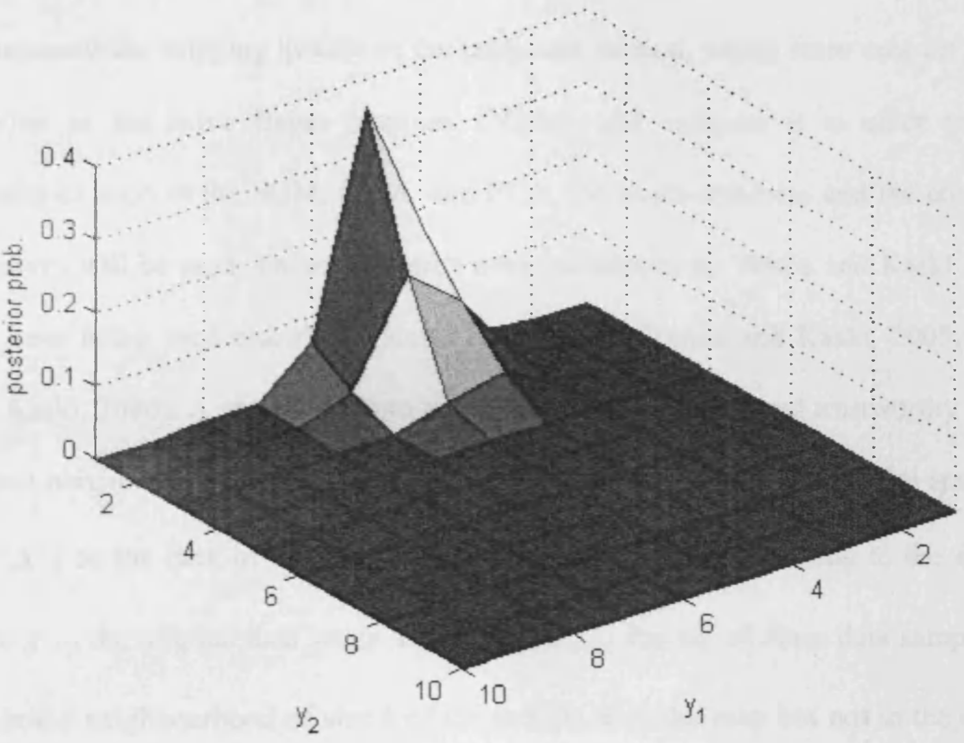

Figure 5.4(c) Posterior probability at epoch=100 for data point number 30 in the Iris data set

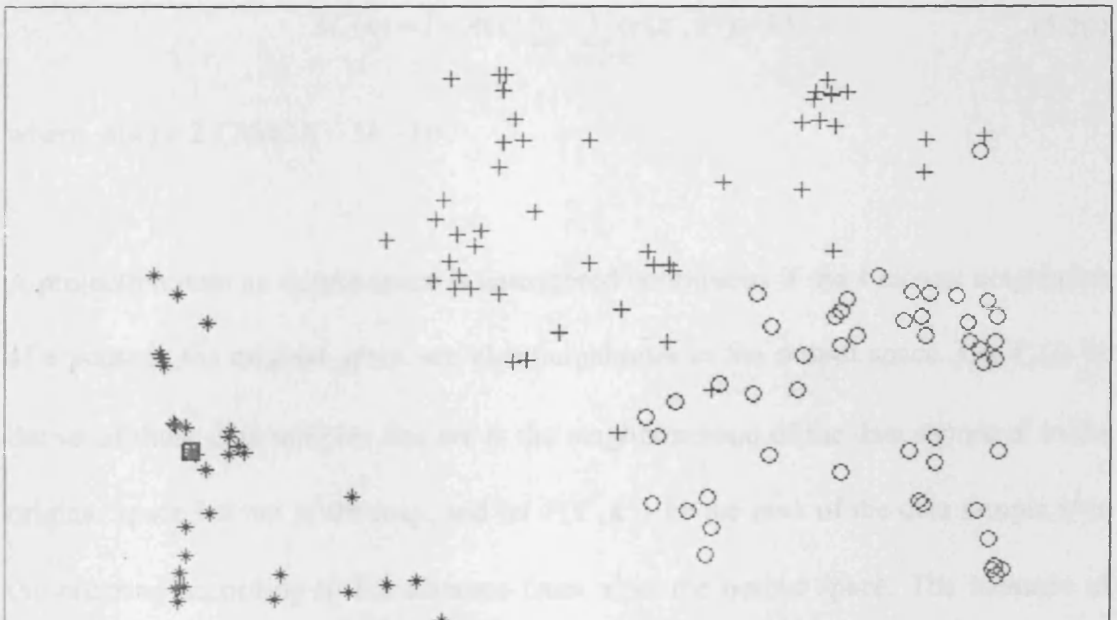

Figure 5.4(d) Output map at epoch=100 with data point 30 represented by a "□".

5.4.3 Comparing map quality with other techniques

To measure the mapping quality of the proposed method, which from now on will be referred as the naive Bayes mapping (NBM), and compare it to other mapping techniques such as the SOM, GTM, and PCA, the trustworthiness and the continuity measures will be used. These measures were introduced by Venna and Kaski (2001), and have being used elsewhere (Kaski *et al.*, 2003; Venna and Kaski, 2005; Venna and Kaski, 2006). A projection onto an output space is considered trustworthy if the k closest neighbours of a point on the map are also neighbours in the original space. Let $r(\mathbf{x}^i,\mathbf{x}^j)$ be the rank of the data sample $\mathbf{x}^j$ in the ordering according to the distance from $\mathbf{x}^i$ in the original data space. Denote by $U_k(i)$ the set of those data samples that are in the neighbourhood of size k of the sample $\mathbf{x}^i$ in the map but not in the original data space. The measure of trustworthiness of the visualisation is defined by

$$M_1(k)=1-A(k)\sum_{i=1}^{N}\sum_{j\in U_k(i)}(r(\mathbf{x}^i,\mathbf{x}^j)-k), \tag{5.20}$$

where $A(k)=2/(Nk(2N-3k-1))$.

A projection onto an output space is considered continuous if the k closest neighbours of a point in the original space are also neighbours in the output space. Let $V_k(i)$ be the set of those data samples that are in the neighbourhood of the data sample $\mathbf{x}^i$ in the original space but not in the map, and let $\hat{r}(\mathbf{x}^i,\mathbf{x}^j)$ be the rank of the data sample $\mathbf{x}^j$ in the ordering according to the distance from $\mathbf{x}^i$ in the output space. The measure of continuity of the visualisation is defined by

$$M_2(k)=1-A(k)\sum_{i=1}^{N}\sum_{j\in V_k(i)}(\hat{r}(\mathbf{x}^i,\mathbf{x}^j)-k). \tag{5.21}$$

In the case of ties in rank ordering, all compatible rank orders are assumed equally likely, and averages of the error measures are computed.

The four projection methods will be tested on 10 benchmark data sets which are described in Table 5.1. For the SOM, GTM, and NBM the output will be a 10x10 grid for all the simulations. For GTM, the number of cycles (epochs) was set to 50, and 10 repetitions where made choosing as the final projection the one that scored the highest log-likelihood, also for each data set, a different number of basis function was tested in order to find the best projection. For these data set, the best projections where found usually for a number of basis functions equal to 16 o 25. For the SOM, the number of epochs used was 1000, and 10 repetitions were made choosing as final projection the one that visually achieved better separation amongst the different classes. The SOM used the neighbourhood function defined in (5.12) and the range defined in (5.19). The learning rate $\alpha(e)$ for epoch e used in these simulations was

$$\alpha(e) = \frac{\alpha_0}{1+\alpha_1 e}, \tag{5.22}$$

with $\alpha_0 = 0.5$ and $\alpha_1 = 0.02$.

The number of epochs for the NBM was set to 50 and 10 repetitions where made choosing as the final projection the one that scored the highest log-likelihood just like the GTM. For better convergence, the learning rate (5.22) was introduced in the updating of $\boldsymbol{\sigma}_k^2$ in (5.16), becoming

$$\boldsymbol{\sigma}_k^{2(t+1)} = \boldsymbol{\sigma}_k^{2(t)} + \alpha(e) f_k^{(t)}\left((\mathbf{x}^{(t+1)} - \boldsymbol{\omega}_k^{(t)})^2 - (1 - f_k^{(t)})\boldsymbol{\sigma}_k^{2(t)}\right), \tag{5.23}$$

with $\alpha_0 = 0.1$ and $\alpha_1 = 0.05$.

The three mapping techniques used a random initialisation for their parameters and each data set was normalised so that each feature had mean = 0 and variance = 1.

The output mapping as well as the trustworthiness and continuity values for each data set are shown in Fig 5 to Fig 14. In what follows, an analysis and discussion for each data set is presented.

The Australian data set (Fig. 5.5(a) – Fig. 5.5(f)). The PCA projection (Fig. 5.5(a)) shows a dense cloud of points where in the centre both classes overlap. The SOM projection (Fig. 5.5(b)), although you can identify regions which contain points from a single class, in general the output presents significant overlapping amongst classes. The GTM projection (Fig. 5.5(c)), due to its continuous (soft) output, presents a less dense projection, where larger regions for each class can be identified. The NBM projection (Fig. 5.5(d)), also has a soft output which makes the identification of regions for each class easier. Notice how the "+" class is mostly projected in the lower left part of the projection while the "O" class is mostly in the upper right side of the map. Analysing the Trustworthiness measure M_1 (Fig. 5.5(e)), the PCA projection neighbourhood preservation is very weak. The SOM achieves the best trustworthiness, the NBM Outperforms the GTM for the first 15 neighbours. These last three projection methods have $M_1 > 0.9$ for up to 15 neighbouring points in the output space. For the Continuity measure M_2 (Fig. 5.5(f)), the PCA projection obtains the worst result, and then it follows slightly better the SOM, GTM, and the NBM. These last three are similar for k =5, and the GTM and NBM are similar for k =15 and k =25.

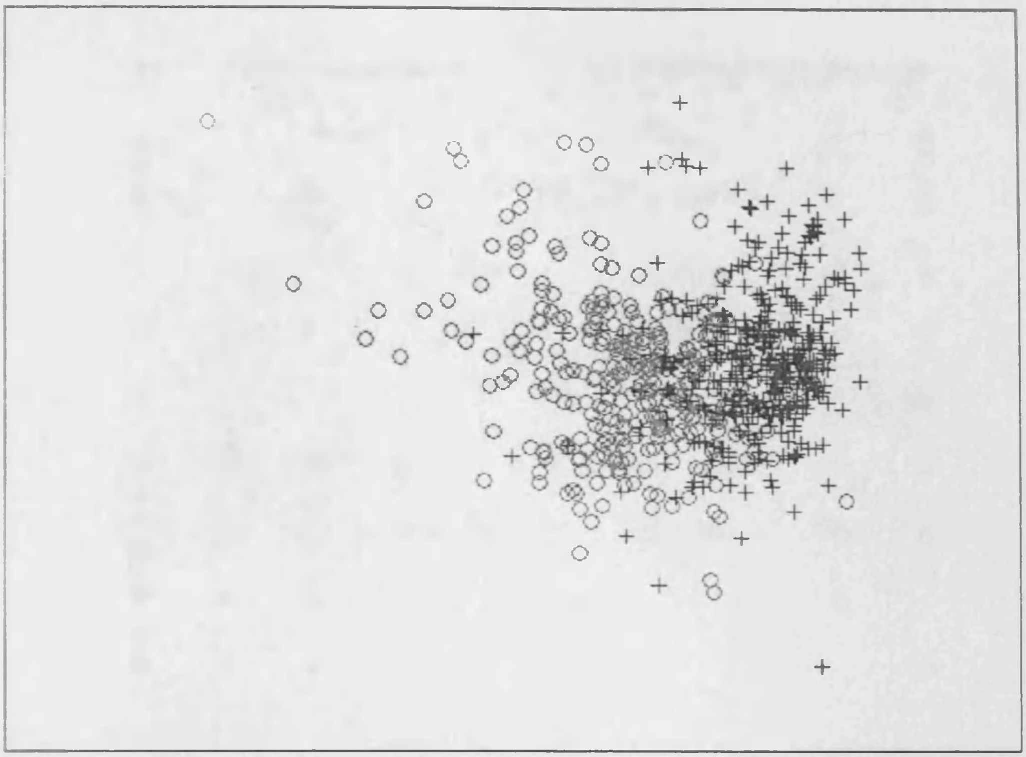

Figure 5.5(a) Australian data set projection using PCA

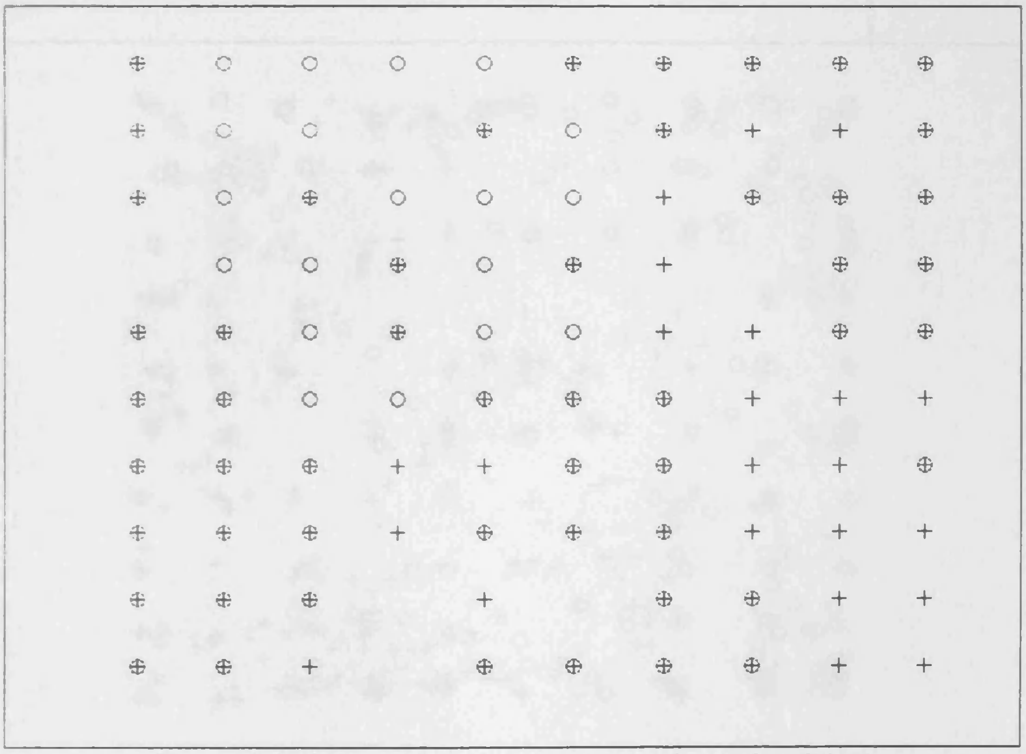

Figure 5.5(b) Australian data set projection using SOM

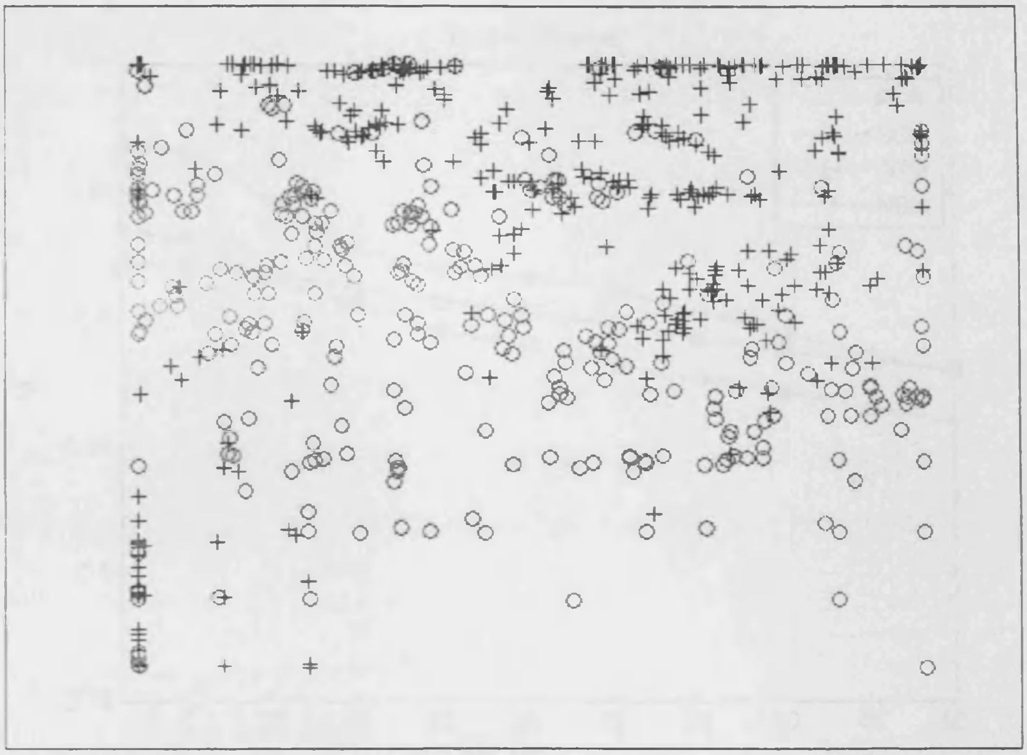

Figure 5.5(c) Australian data set projection using GTM

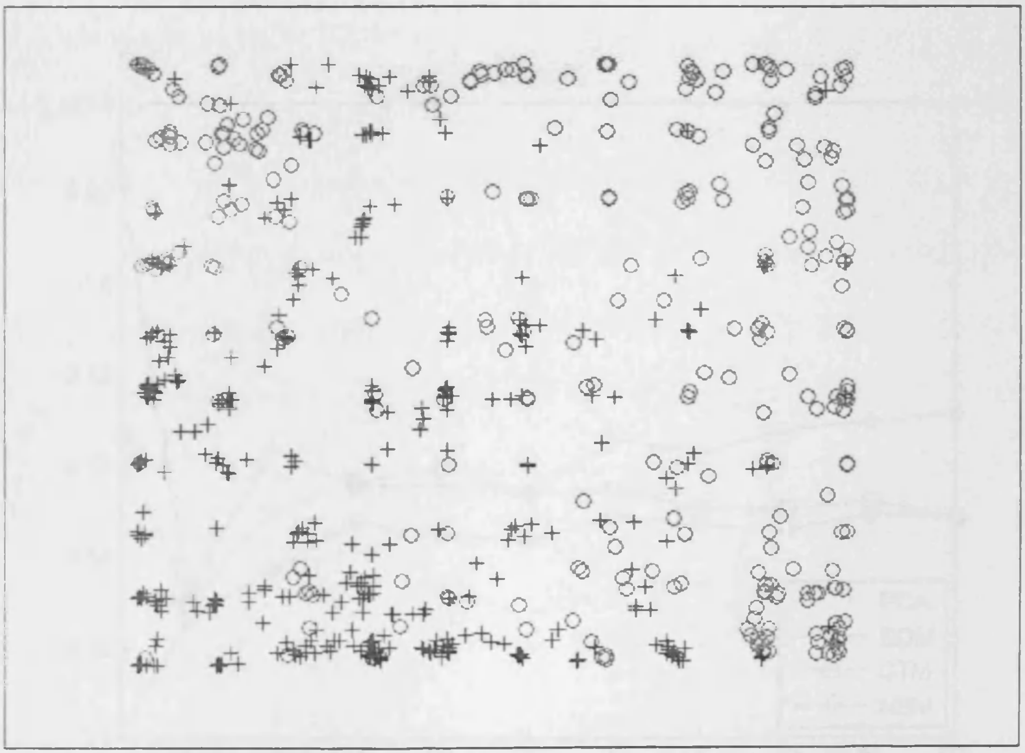

Figure 5.5(d) Australian data set projection using NBM

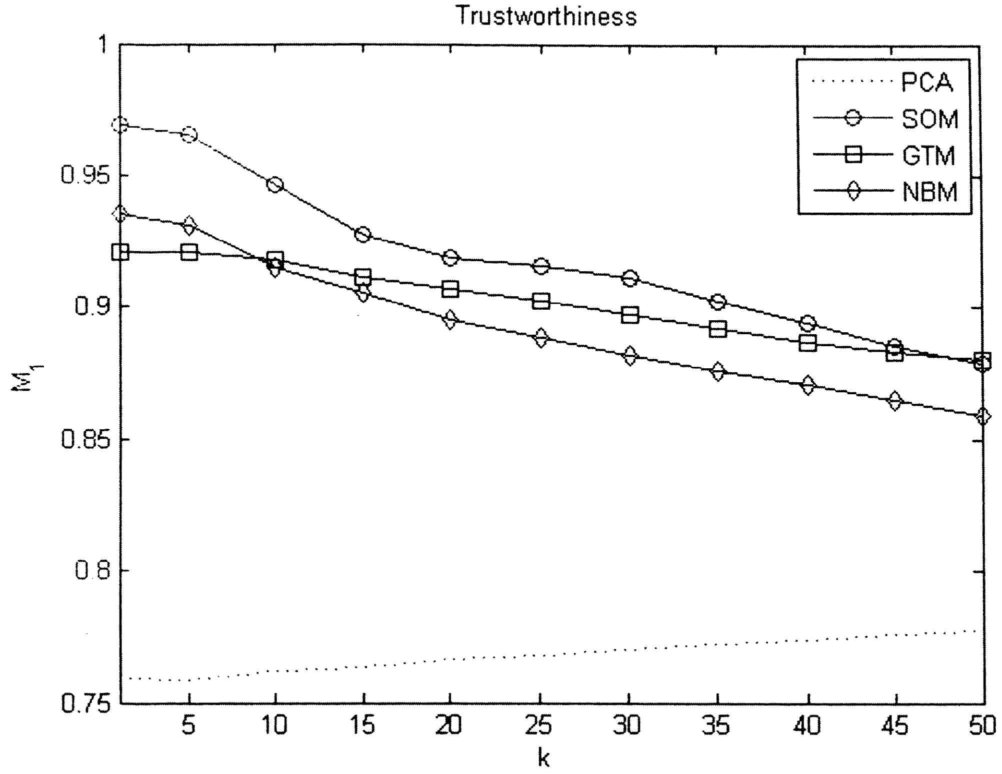

Figure 5.5(e) Trustworthiness measure for the Australian data set projection

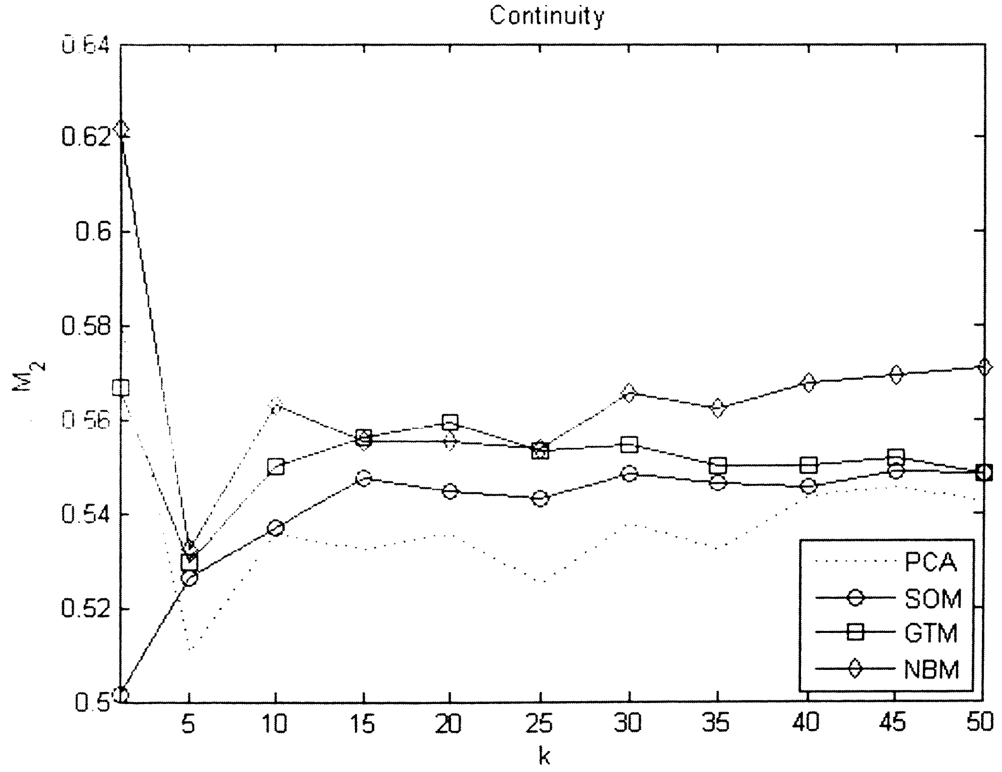

Figure 5.5(f) Continuity measure for the Australian data set projection

The Breast data set (Fig. 5.6(a) – Fig. 5.6(f)). The PCA projection (Fig. 5.6(a)) shows two dense clouds of points distributed along a line with two outliers from class "+", also between these two clouds of points there is a slight overlapping between points from different classes. The SOM projection (Fig. 5.6(b)), clearly separates the two classes in two different regions of the map, "+" on the left side and "O" on the right, with a few neurons in the middle containing points from both classes. The GTM projection (Fig. 5.6(c)), presents a dense projection for the "+" class on the left side and a sparse projection of the "O" class on the right side. The NBM projection (Fig. 5.6(d)), also has a dense projection for the "+" class located in the lower part of the projection and a sparse projection for the "O" class located in the upper part of the mapping. In both cases, GTM and NBM, the dense and sparse projection makes it simple to identify each class. Analysing the Trustworthiness measure M_1 (Fig. 5.6(e)), the PCA projection obtains the lowest values although $M_1>0.85$ for *k* up to 10 which is not bad. The SOM achieves the best trustworthiness, and then follows the GTM and finally the NBM. The NBM achieves similar trustworthiness with GTM, for *k* up to 5, and the three projection methods have $M_1 > 0.9$ for up to 10 neighbouring points in the output space. For the Continuity measure M_2 (Fig. 5.6(f)), the PCA projection obtains the best result, and then it follows the NBM, the SOM, and finally the GTM. The PCA and NBM are similar for *k* up to 10.

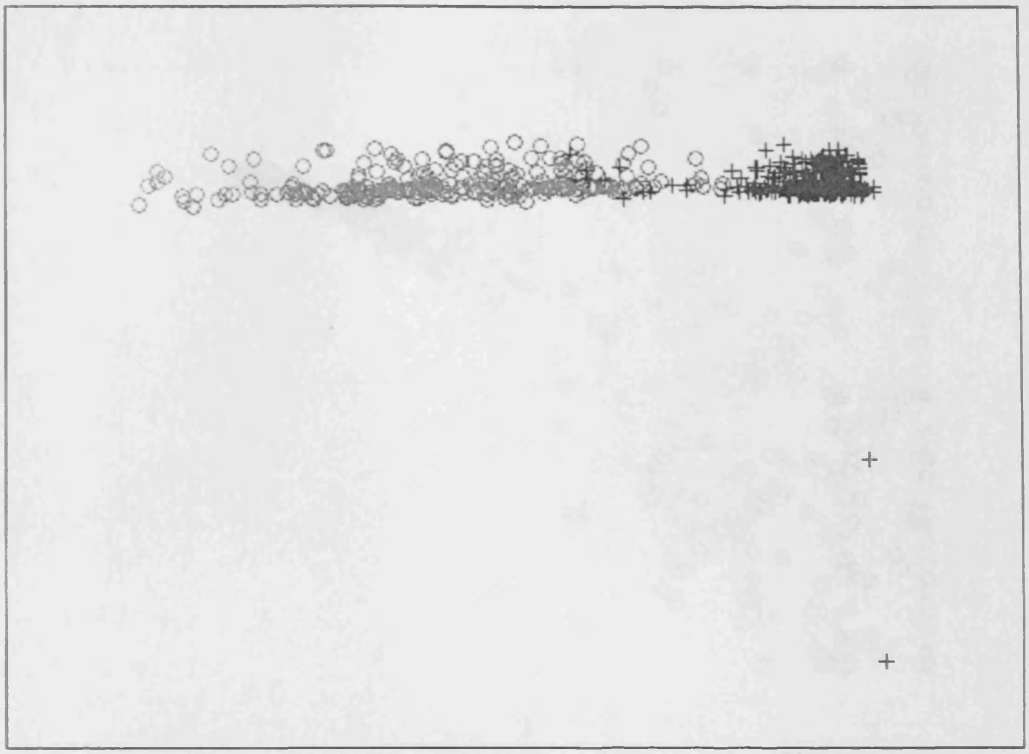

Figure 5.6(a) Breast data set projection using PCA

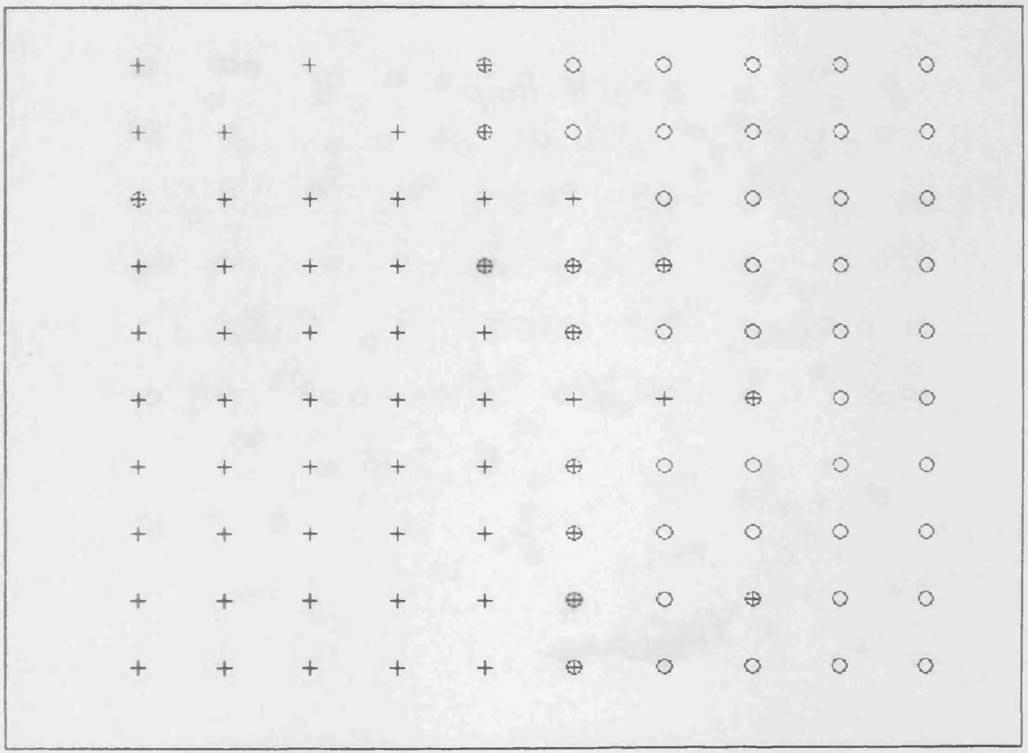

Figure 5.6(b) Breast data set projection using SOM

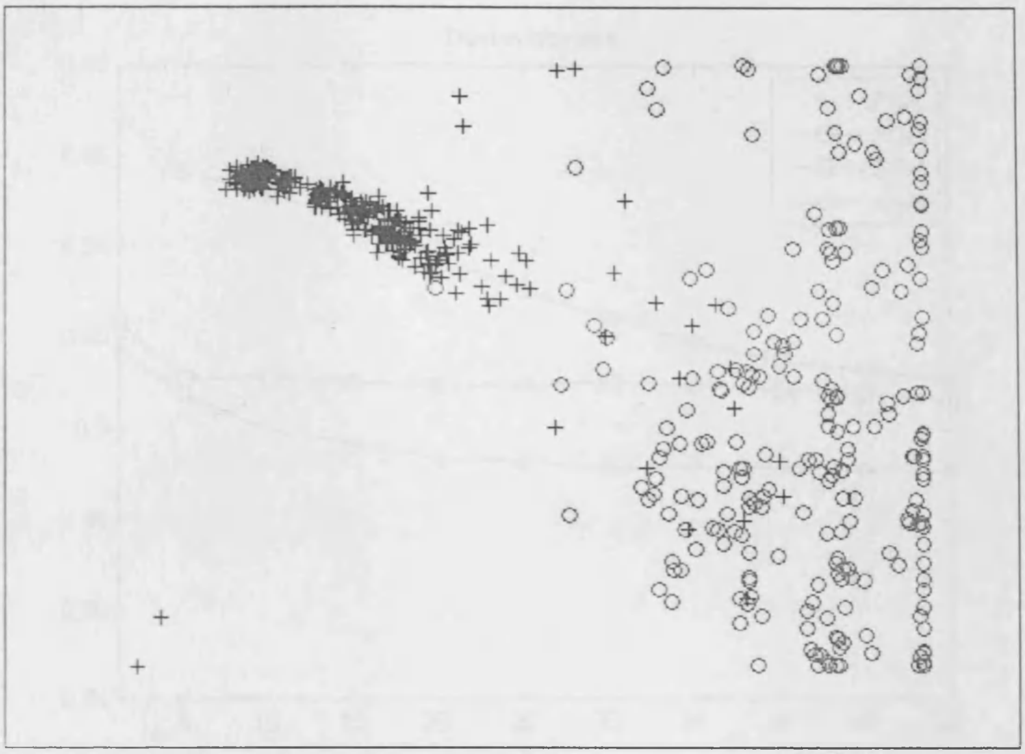

Figure 5.6(c) Breast data set projection using GTM

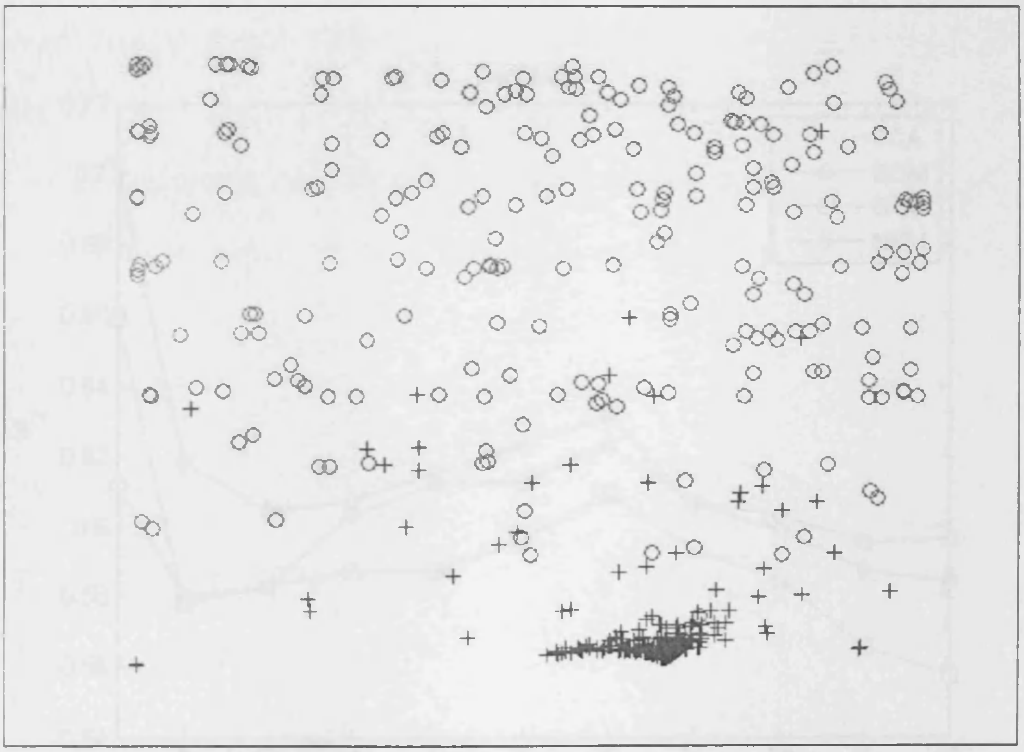

Figure 5.6(d) Breast data set projection using NBM

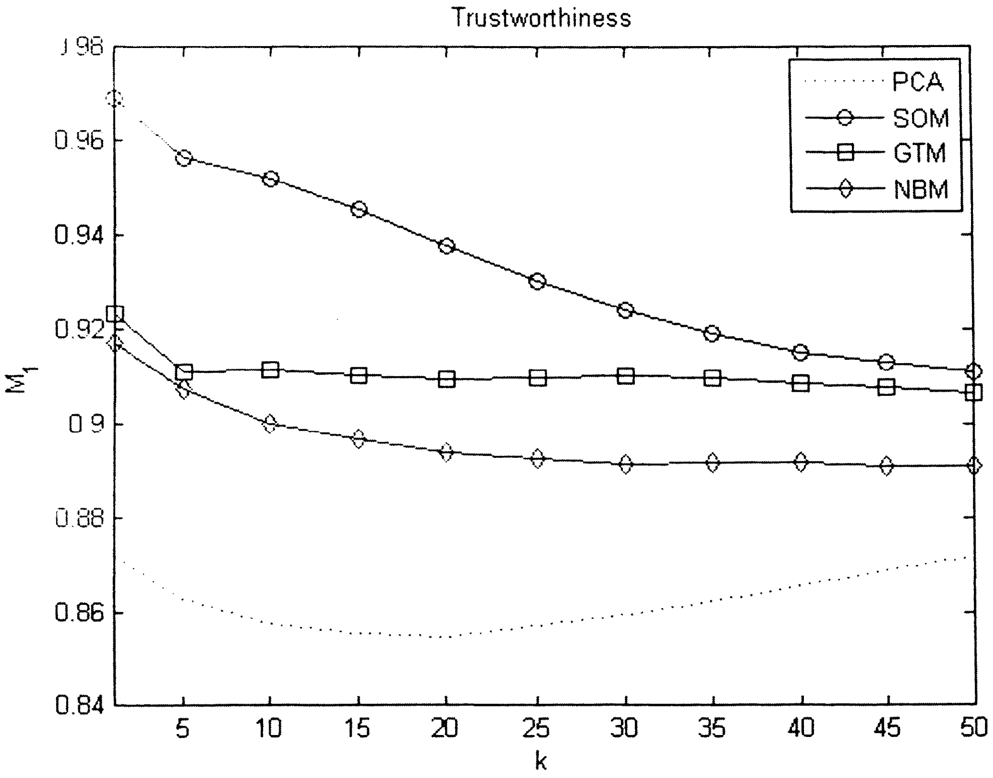

Figure 5.6(e) Trustworthiness measure for the Breast data set projection

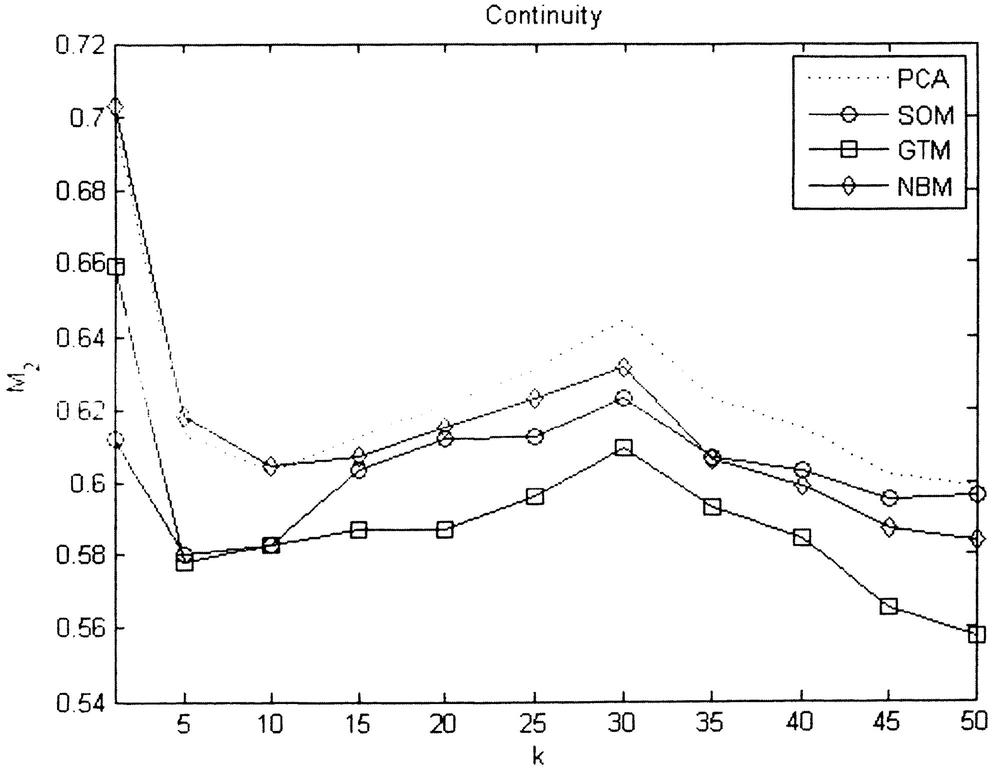

Figure 5.6(f) Continuity measure for the Breast data set projection

The Cleve data set (Fig. 5.7(a) – Fig. 5.7(f)). The PCA projection (Fig. 5.7(a)) shows a less dense cloud of points, where the "+" is located on the right side of the cloud of points and the "O" class is on the left side, with some overlapping in the middle. The SOM projection (Fig. 5.7(b)), separates the two classes in two different regions of the map, "+" on the top right side and "O" on the bottom left, with several neurons, in different regions of the map, containing points from both classes. The GTM projection (Fig. 5.7(c)), presents a sparse projection, where the "+" class is mostly concentrated on the bottom side of the map and the "O" class on the top side. The NBM projection (Fig. 5.7(d)), also has a sparse projection concentrating the majority of the "+" class on the upper right triangle and the "O" class in the lower left triangle. Analysing the Trustworthiness measure M_1 (Fig. 5.7(e)), the PCA projection obtains the lowest values with $M_1<0.8$ for k up to 25. The SOM achieves the best trustworthiness for k up to 20, then, it drops significantly. The GTM and the NBM are similar for k up to 15, then, GTM obtains slightly better values than the NBM. For the Continuity measure M_2 (Fig. 5.7(f)), the NBM projection obtains the best result, and then it follows the GTM, the PCA, and finally the SOM. The NBM and SOM are similar for k =5.

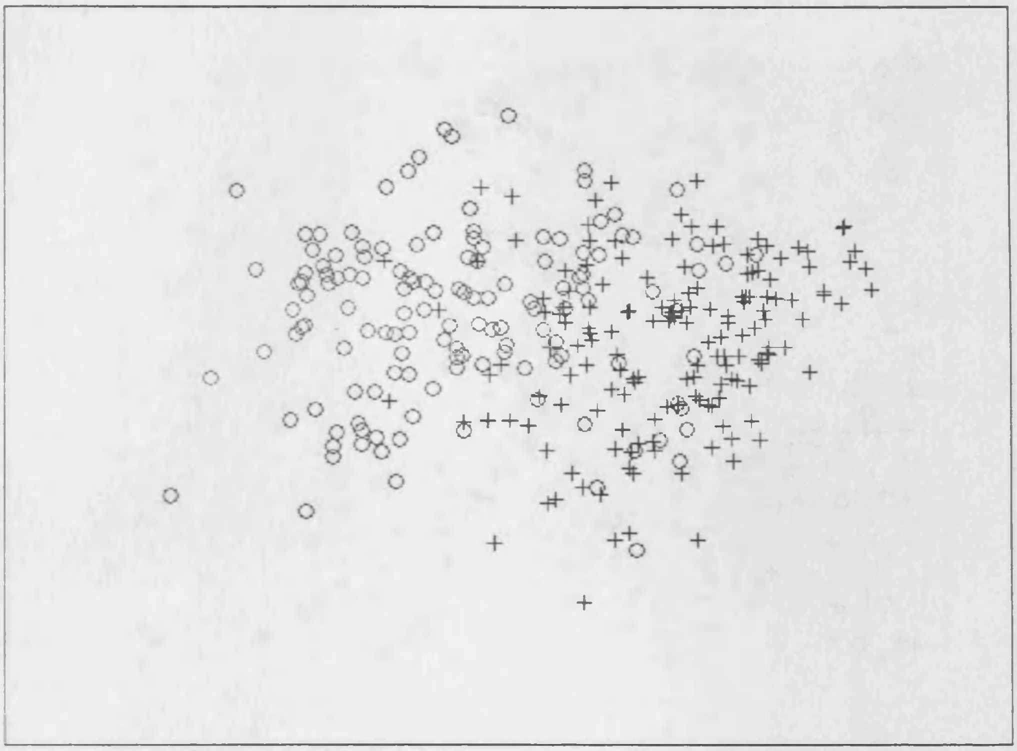

Figure 5.7(a) Cleve data set projection using PCA

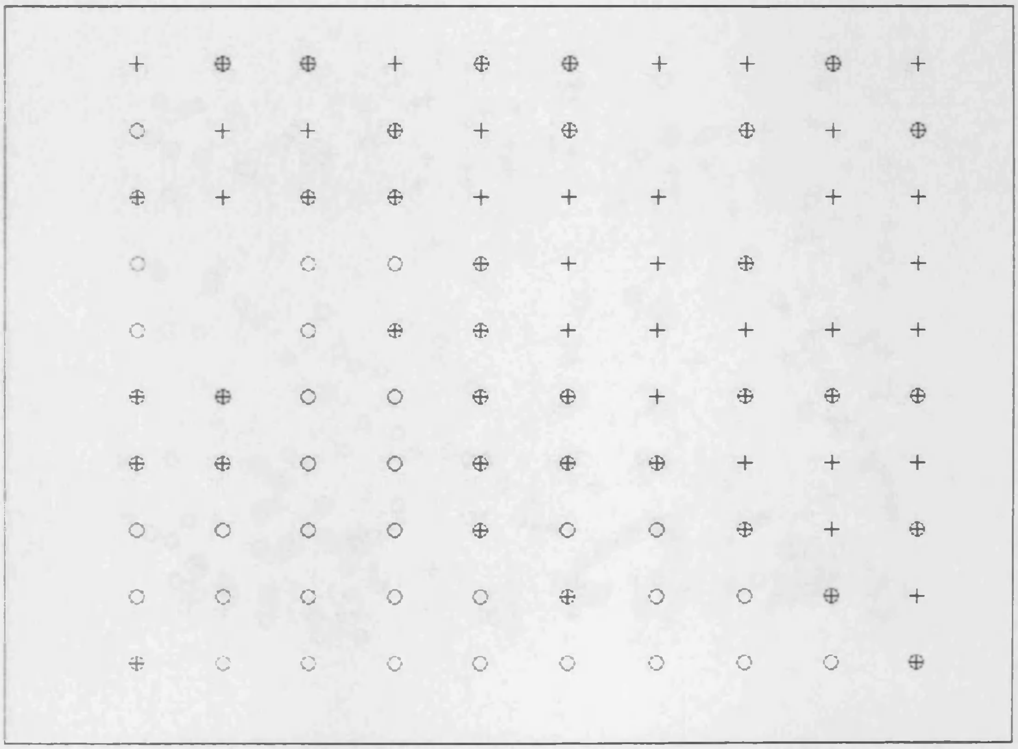

Figure 5.7(b) Cleve data set projection using SOM

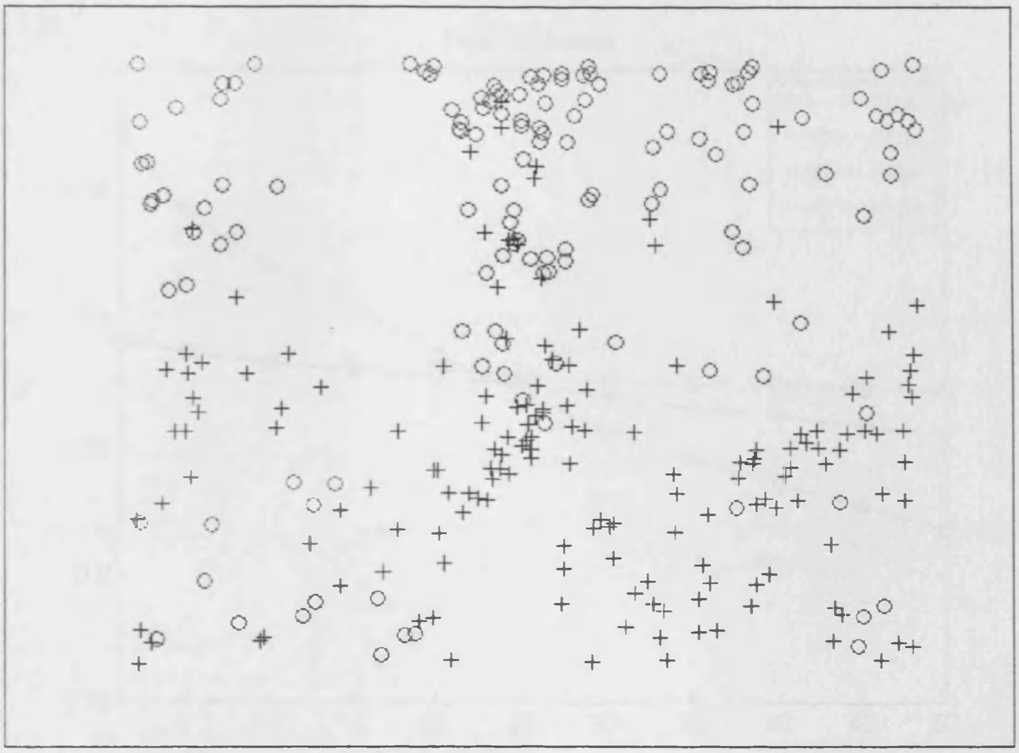

Figure 5.7(c) Cleve data set projection using GTM

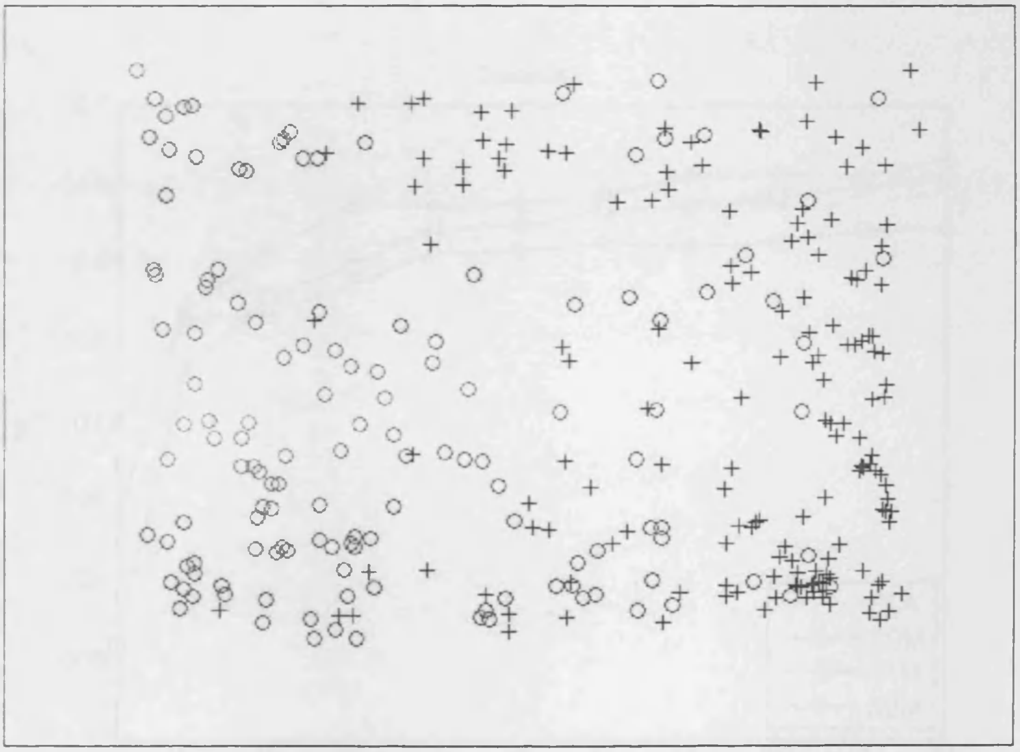

Figure 5.7(d) Cleve data set projection using NBM

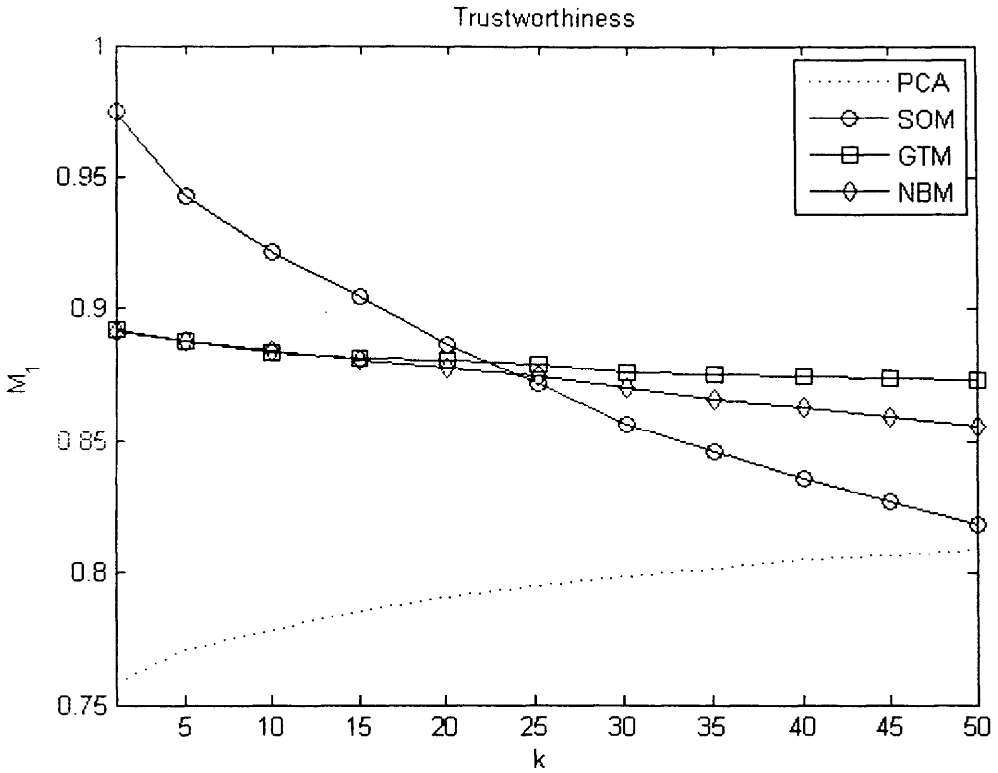

Figure 5.7(e) Trustworthiness measure for the Cleve data set projection

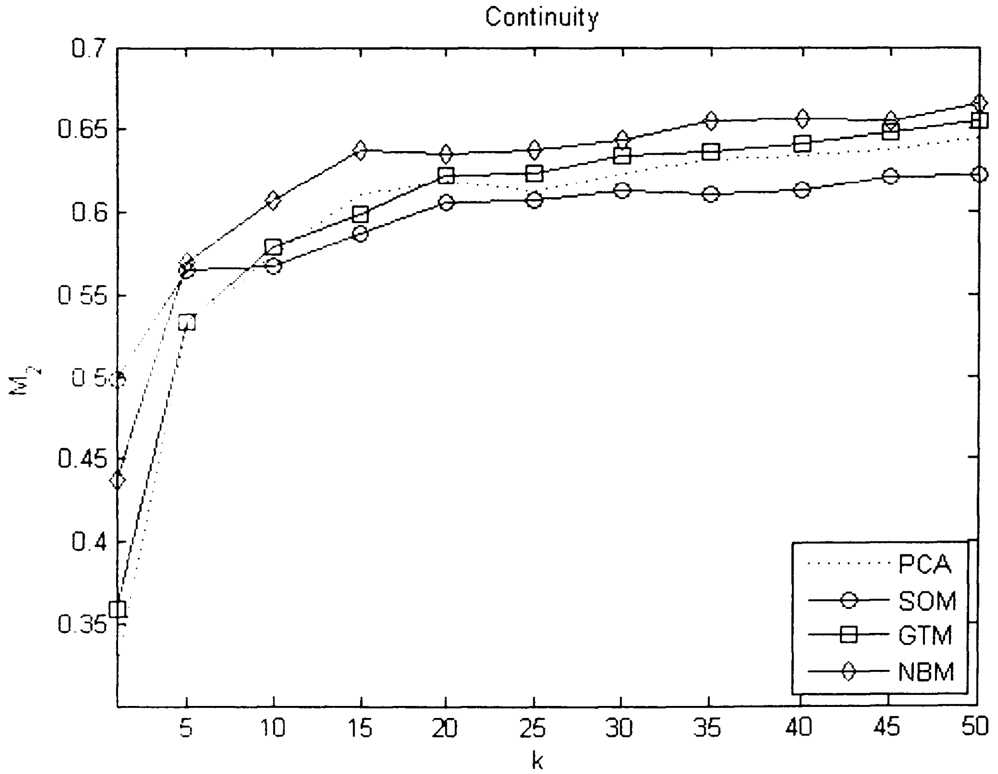

Figure 5.7(f) Continuity measure for the Cleve data set projection

The Crx data set (Fig. 5.8(a) – Fig. 5.8(f)). The PCA projection (Fig. 5.8(a)) shows two clouds of points, one bigger than the other. Each could of points contain dense and sparse regions as well as points from both classes. In each cloud, the "O" class is mostly concentrated on the left side, and the "+" class on the right side. The SOM projection (Fig. 5.8(b)), although regions which contain points from a single class can be identified, in general, the output presents significant overlapping amongst classes. The GTM projection (Fig. 5.8(c)), presents a sparse projection of the two clouds found in the PCA projection, where the "+" class is mostly concentrated on the left side of and the "O" class on the right side, of each cloud. The NBM projection (Fig. 5.8(d)), also has a sparse projection of the two clouds which appeared in the PCA mapping. The big cloud is on the left side, where the "+" class is mostly on the upper left region and the "O" class in the lower part of the mapping. The small cloud is located in the top right corner, following a similar class distribution as the description of the big cloud of points. Analysing the Trustworthiness measure M_1 (Fig. 5.8(e)), the PCA projection obtains the lowest values with $0.84<M_1<0.86$ for all k computed, which is not bad. The SOM achieves the best trustworthiness for k up to 15, then, it drops. At k =10, the NBM equals the SOM and then outperforms the SOM for the rest of the k values. The NBM obtains better values of M_1 than the GTM for up to k =20, and then onwards they perform similar. For the Continuity measure M_2 (Fig. 5.8(f)), the PCA projection obtains the best result for up to k =10, and then it follows the NBM, the GTM, and finally the SOM. Although for the four projection methods the overall values of M_2 were quite low.

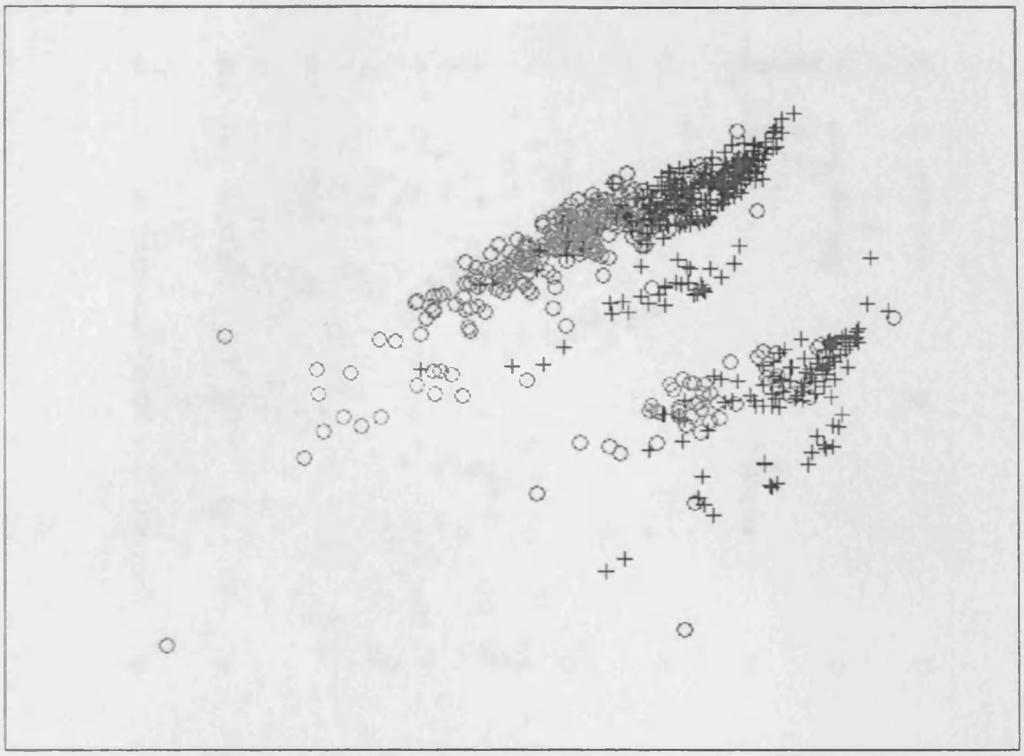

Figure 5.8(a) Crx data set projection using PCA

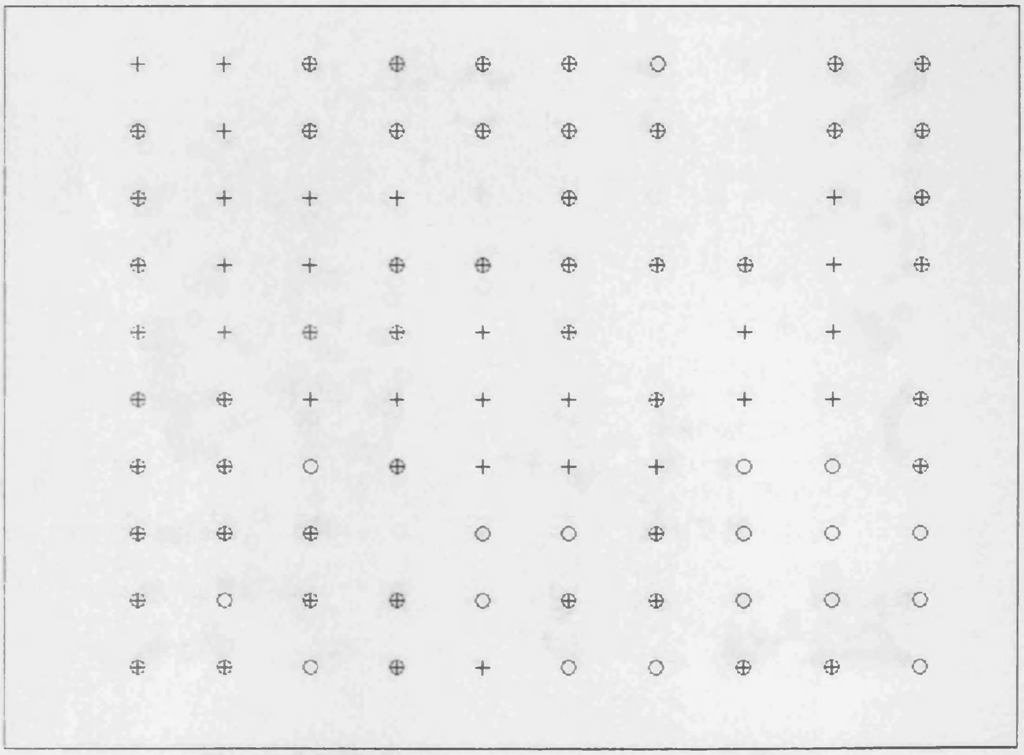

Figure 5.8(b) Crx data set projection using SOM

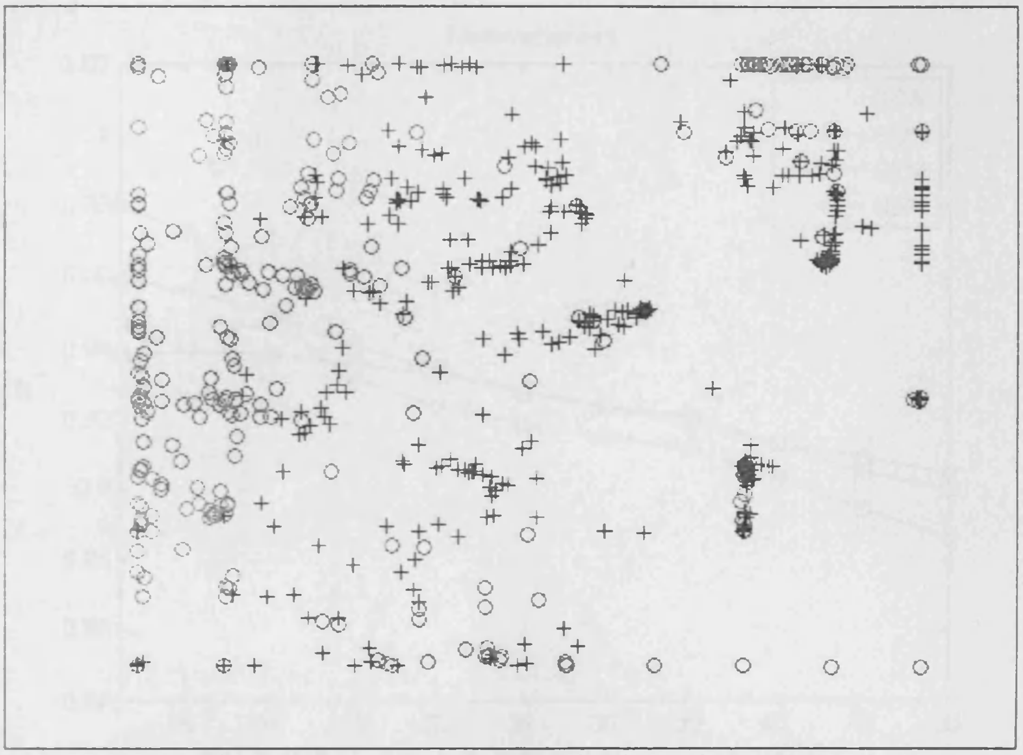

Figure 5.8(c) Crx data set projection using GTM

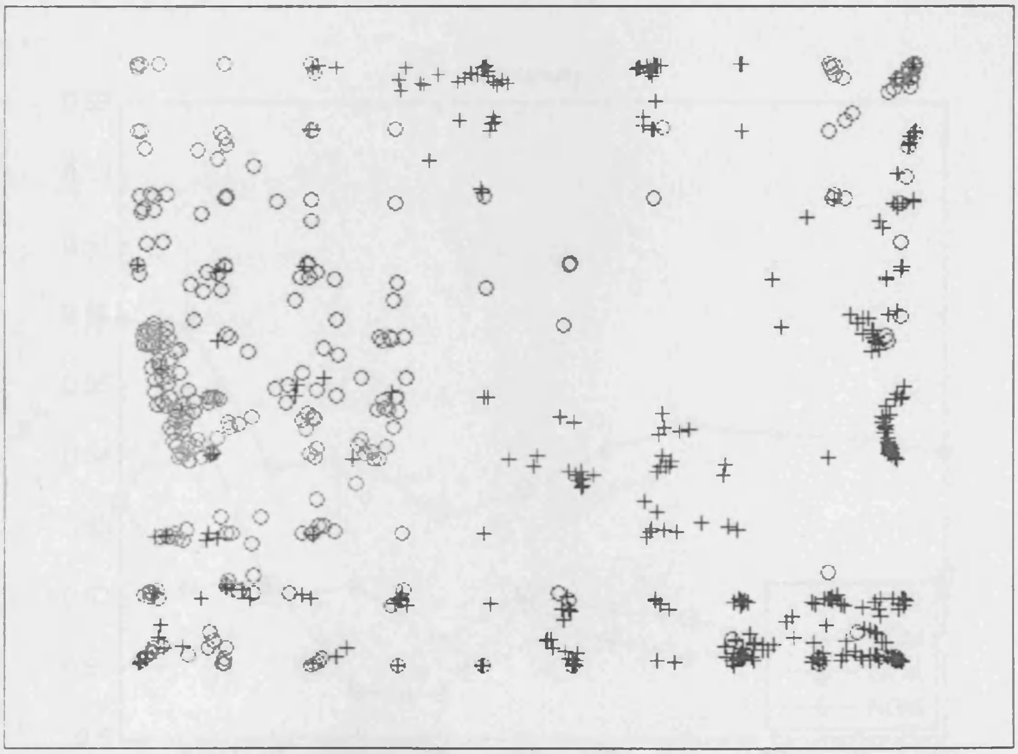

Figure 5.8(d) Crx data set projection using NBM

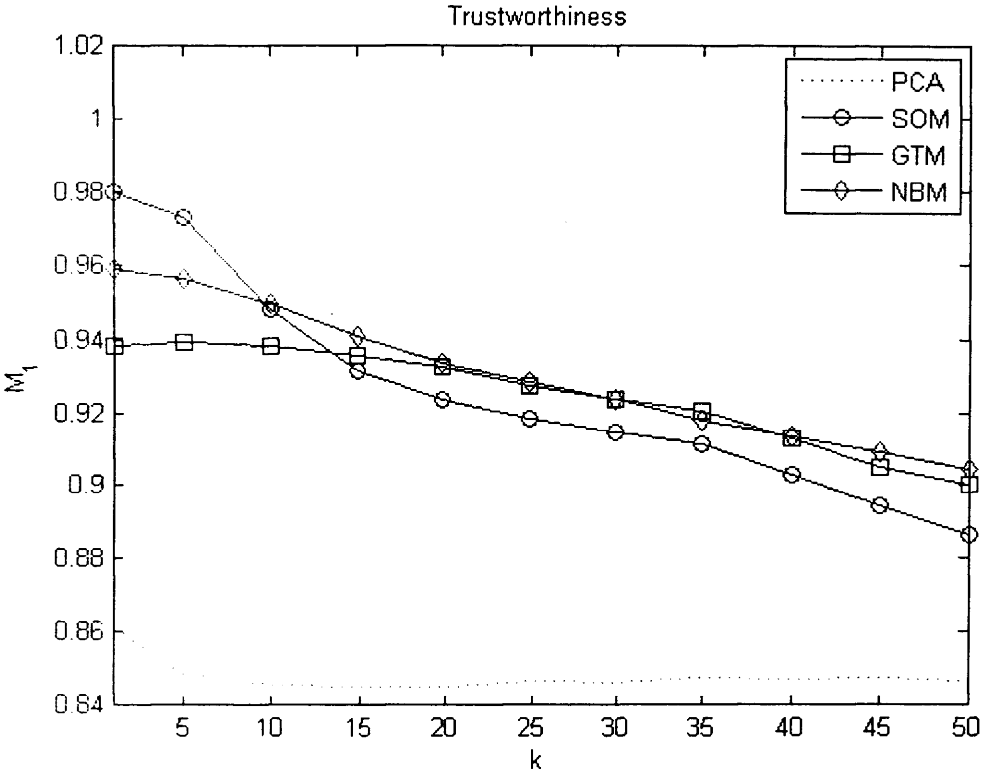

Figure 5.8(e) Trustworthiness measure for the Crx data set projection

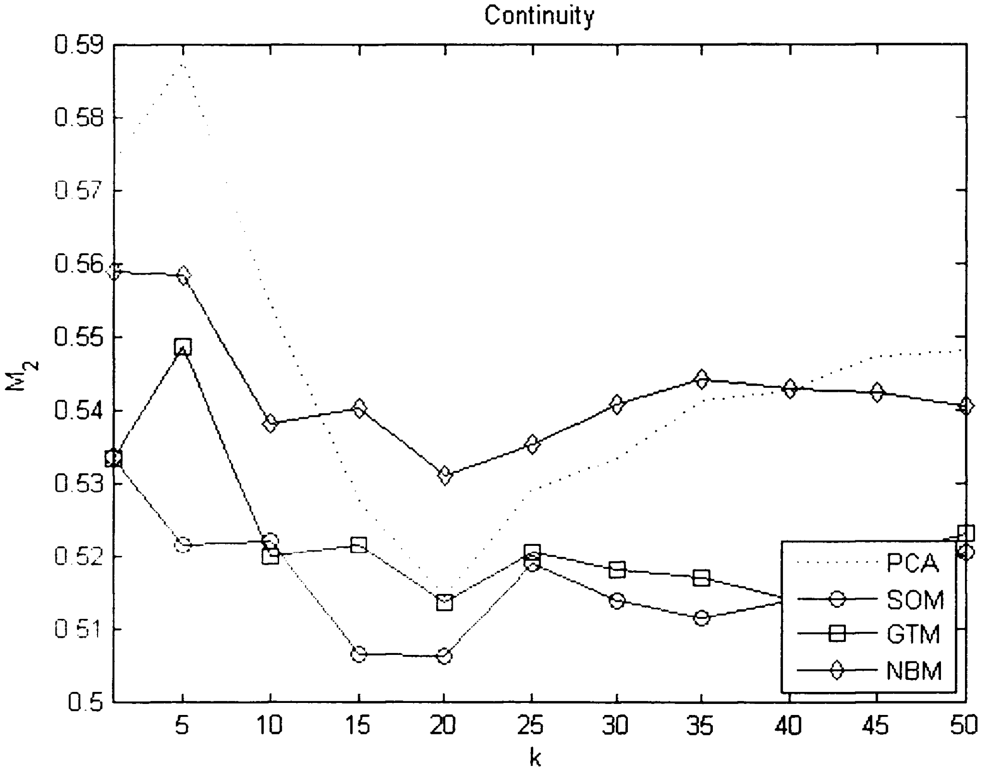

Figure 5.8(f) Continuity measure for the Crx data set projection

The Diabetes data set (Fig. 5.9(a) – Fig. 5.9(f)). The PCA projection (Fig. 5.9(a)) shows a semi-dense cloud of points, the "O" class is mostly concentrated on lower and left side of the mapping, and the "+" class on the top right side, significant overlapping can be appreciated as well. The SOM projection (Fig. 5.9(b)), shows most of its neurons containing points from both classes, only the "+" class has a small distinguishable region in the map while the "O" class has only six isolated neurons. The GTM projection (Fig. 5.9(c)), presents a sparse mapping concentrating the "+" class mostly in the centre and the "O" class in the surroundings of the map. Significant overlapping can be viewed as well. The NBM projection (Fig. 5.9(d)), also has a sparse projection concentrating the "O" class in the lower right triangle of the map, and the "+" class on the upper left triangle. High overlapping regions are also present. Analysing the Trustworthiness measure M_1 (Fig. 5.9(e)), the PCA projection obtains the lowest values with $0.8<M_1<0.84$ for all k computed, which is not bad. The SOM achieves the best trustworthiness for k up to 40, then, it drops slightly. The NBM outperforms the GTM up to k =35, then GTM is slightly better. In general, the three nonlinear projections achieve $M_1>0.9$ for all k. For the Continuity measure M_2 (Fig. 5.9(f)), the PCA projection obtains the best result for all k, and then it follows the NBM, the GTM, and finally the SOM. The NBM and GTM share similar values up to k = 15, then the NBM obtains slightly better results.

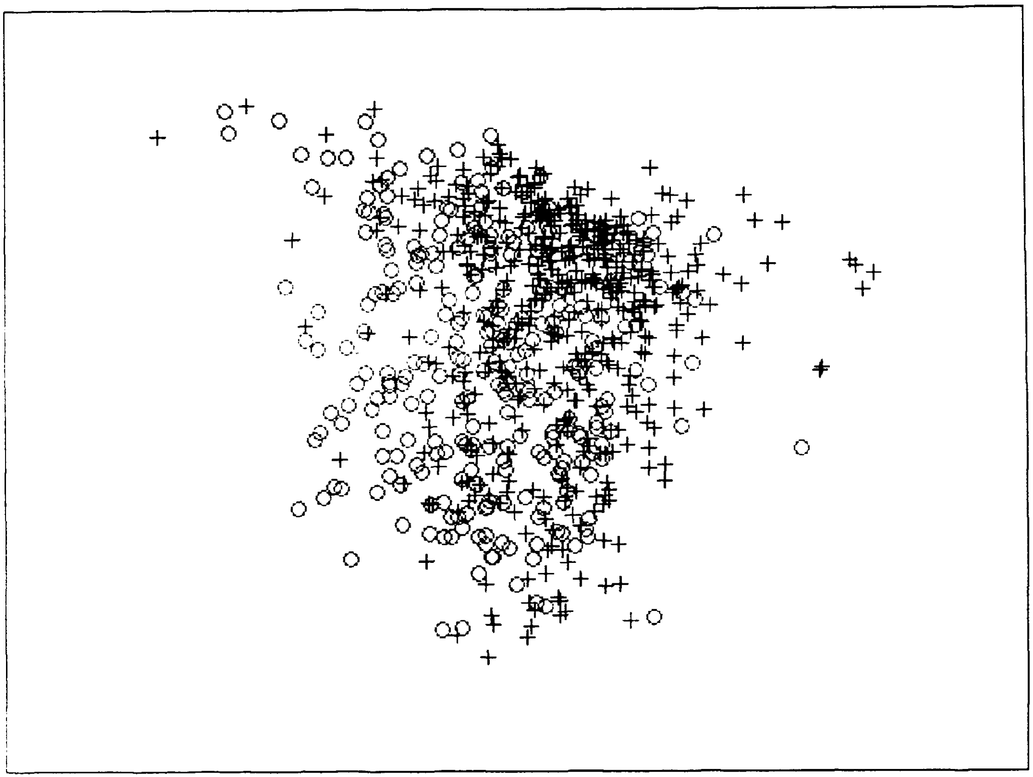

Figure 5.9(a) Diabetes data set projection using PCA

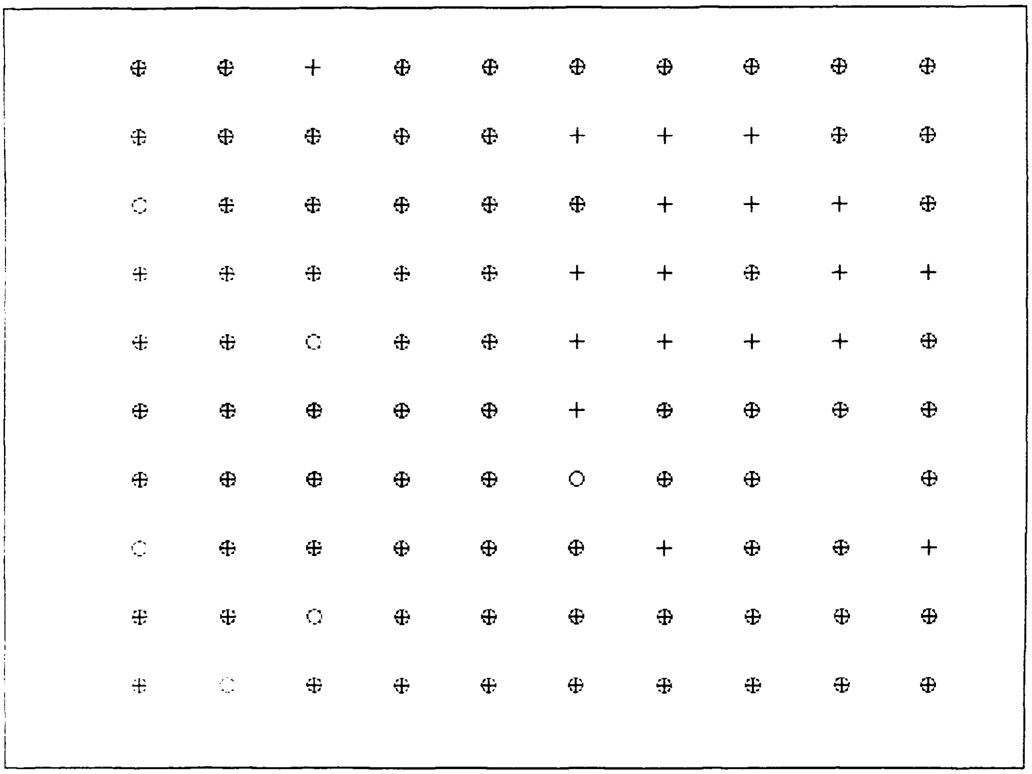

Figure 5.9(b) Diabetes data set projection using SOM

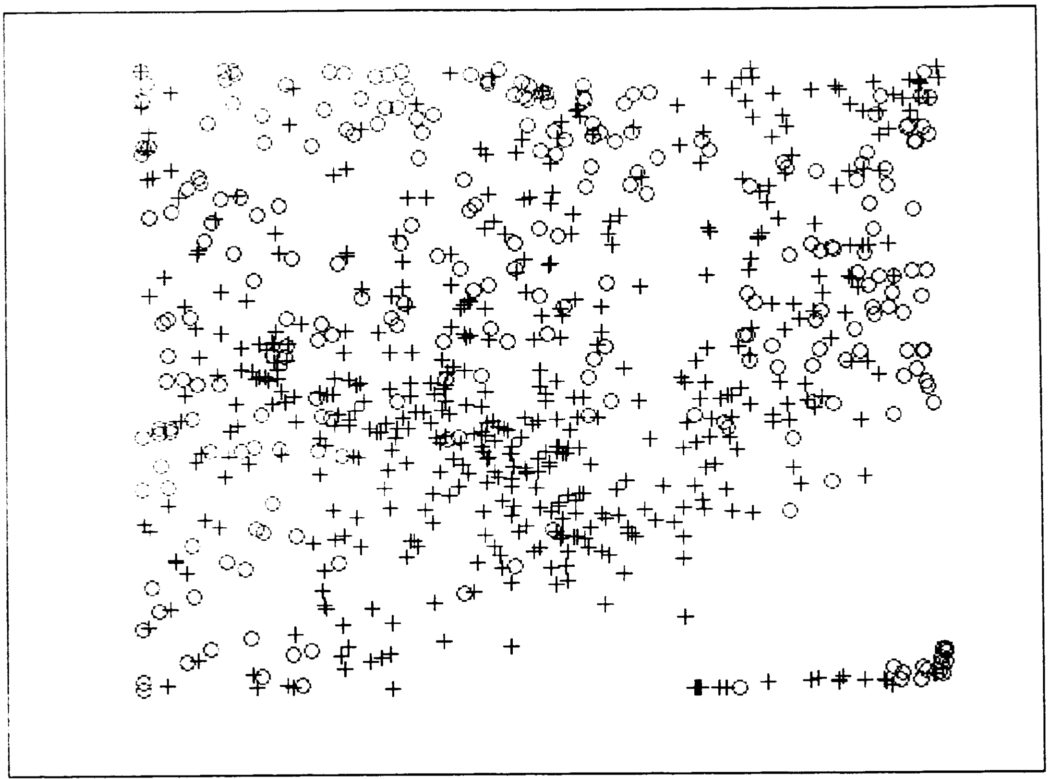

Figure 5.9(c) Diabetes data set projection using GTM

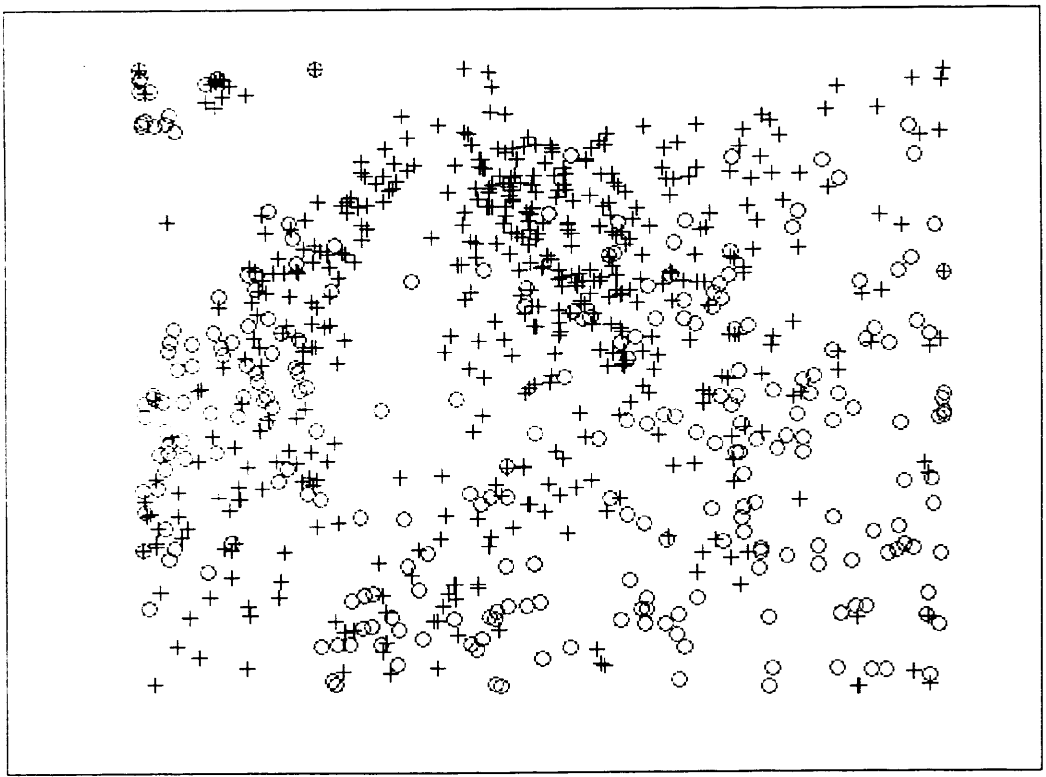

Figure 5.9(d) Diabetes data set projection using NBM

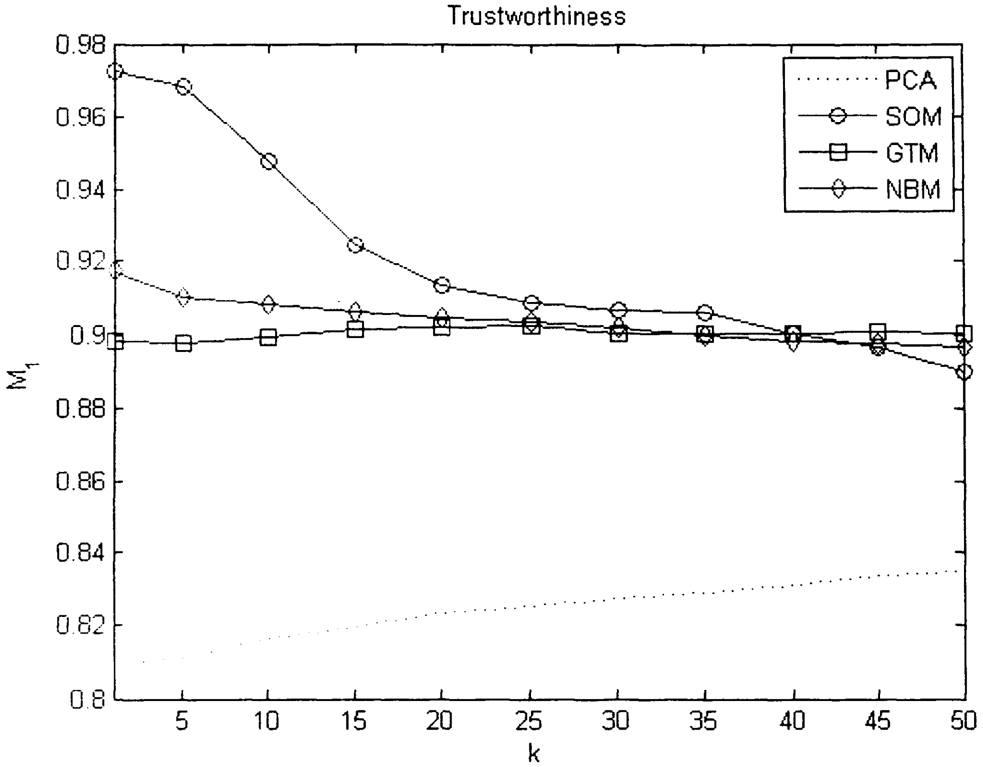

Figure 5.9(e) Trustworthiness measure for the Diabetes data set projection

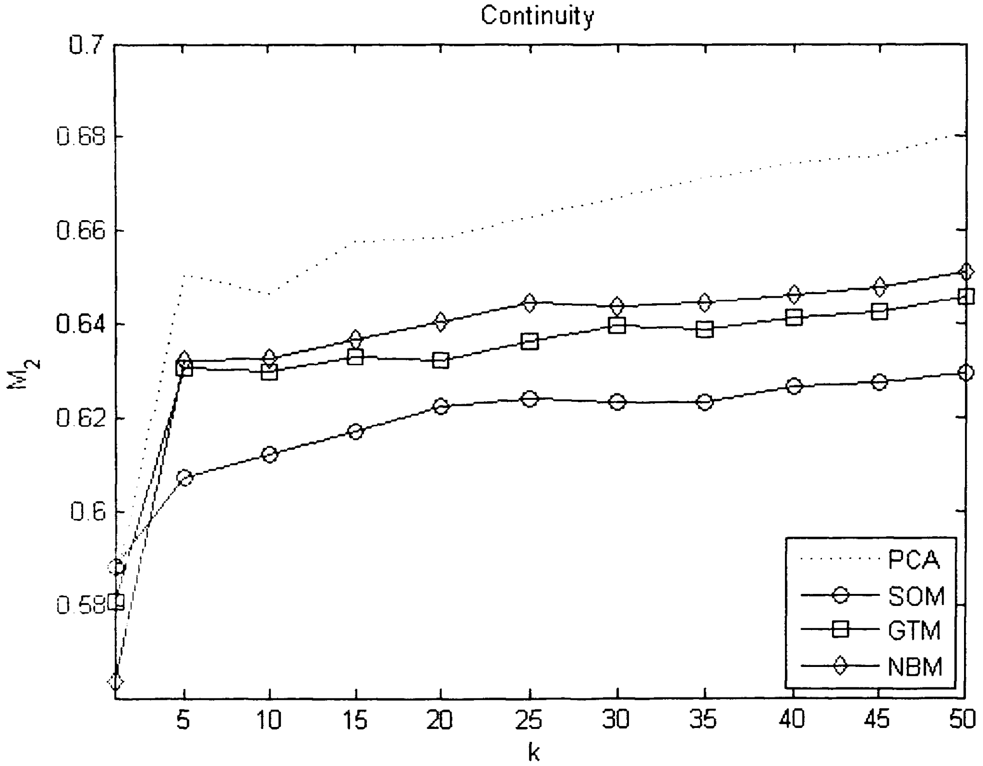

Figure 5.9(f) Continuity measure for the Diabetes data set projection

The Heart data set (Fig. 5.10(a) – Fig. 5.10(f)). The PCA projection (Fig. 5.10(a)) shows a semi-dense cloud of points, the "O" class is mostly concentrated on the left side of the mapping, and the "+" class on the right side. Some overlapping between the two classes can be appreciated, mostly on the right side. The SOM projection (Fig. 5.10(b)), shows the distribution of the two classes relatively clear, the "+" class is located in the lower right triangle in the map, and the "O" class is on the top left triangle. Several neurons containing points from both classes can be seen, especially on the upper left side of the map corresponding to the "O" class. The GTM projection (Fig. 5.10(c)), presents a sparse mapping concentrating the "+" class mostly in the lower left triangle of the mapping and the "O" class in upper right triangle. Some overlapping can be seen in the middle and lower regions of the map. The NBM projection (Fig. 5.10(d)), also has a sparse projection concentrating the "O" class in the upper left region of the map, and the "+" class on the lower right region. Some overlapping where both classes intersect can be seen. Analysing the Trustworthiness measure M_1 (Fig. 5.10(e)), the PCA projection obtains the lowest values with $0.75<M_1<0.82$ for all k computed. The SOM achieves the best trustworthiness for k up to 25, then, it drops considerably. The NBM has a similar value to the GTM for k =5, then GTM is slightly better. In general, the three nonlinear projections achieve $M_1>0.85$ for k up to 30. For the Continuity measure M_2 (Fig. 5.10(f)), the PCA, GTM, and the NBM projections obtained similar values and the three are better than the SOM.

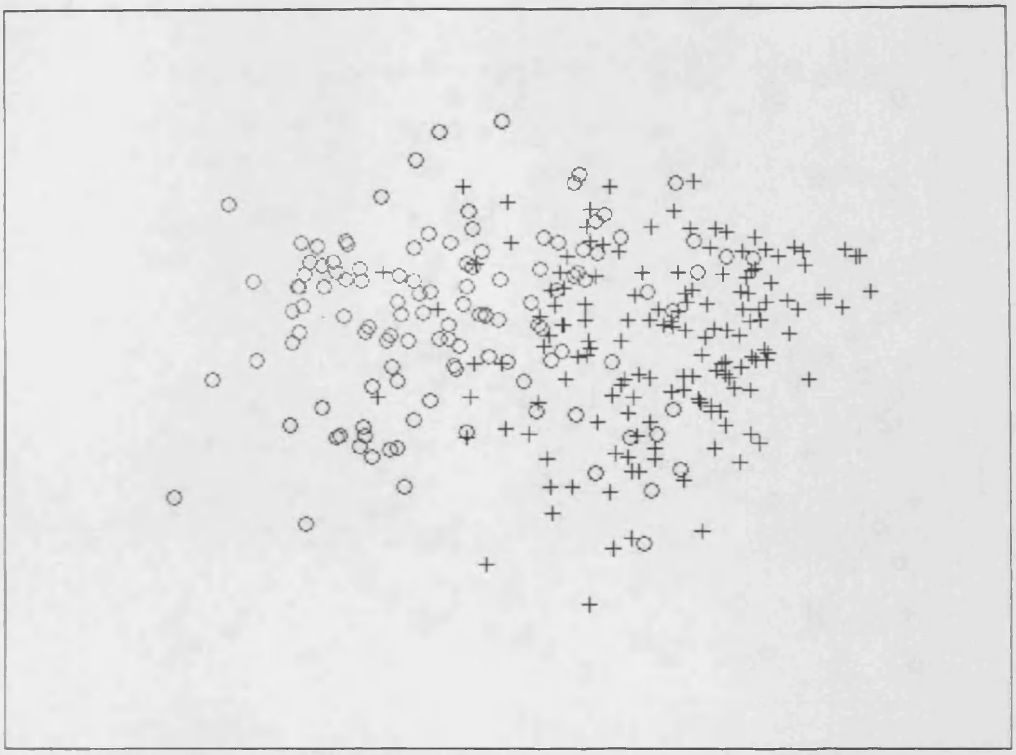

Figure 5.10(a) Heart data set projection using PCA

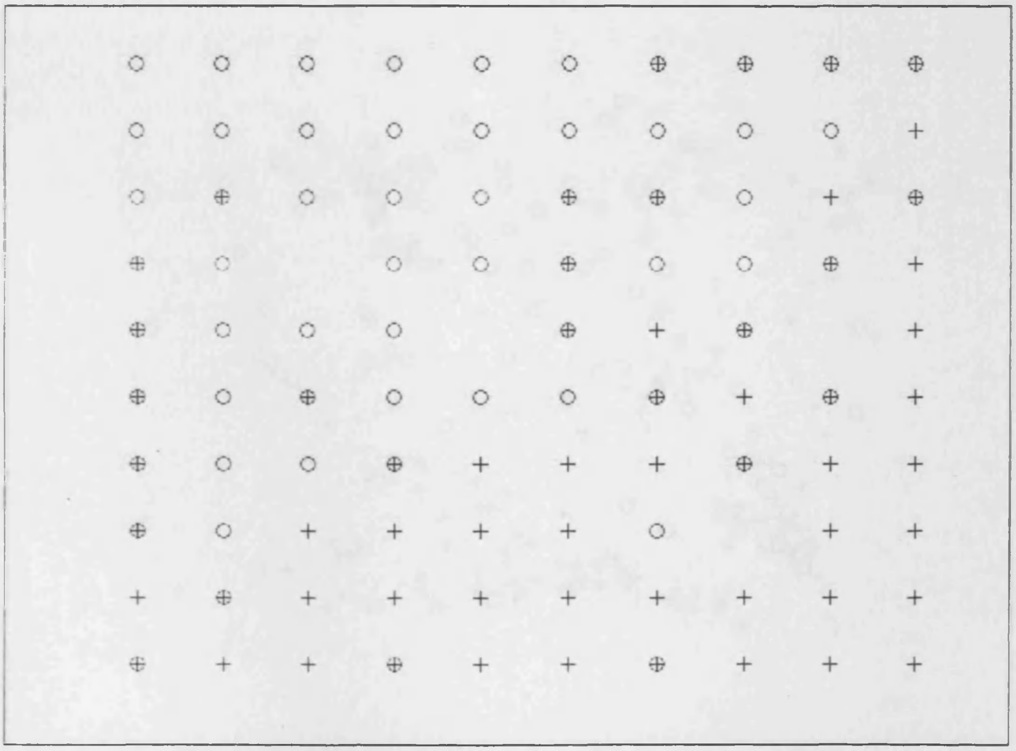

Figure 5.10(b) Heart data set projection using SOM

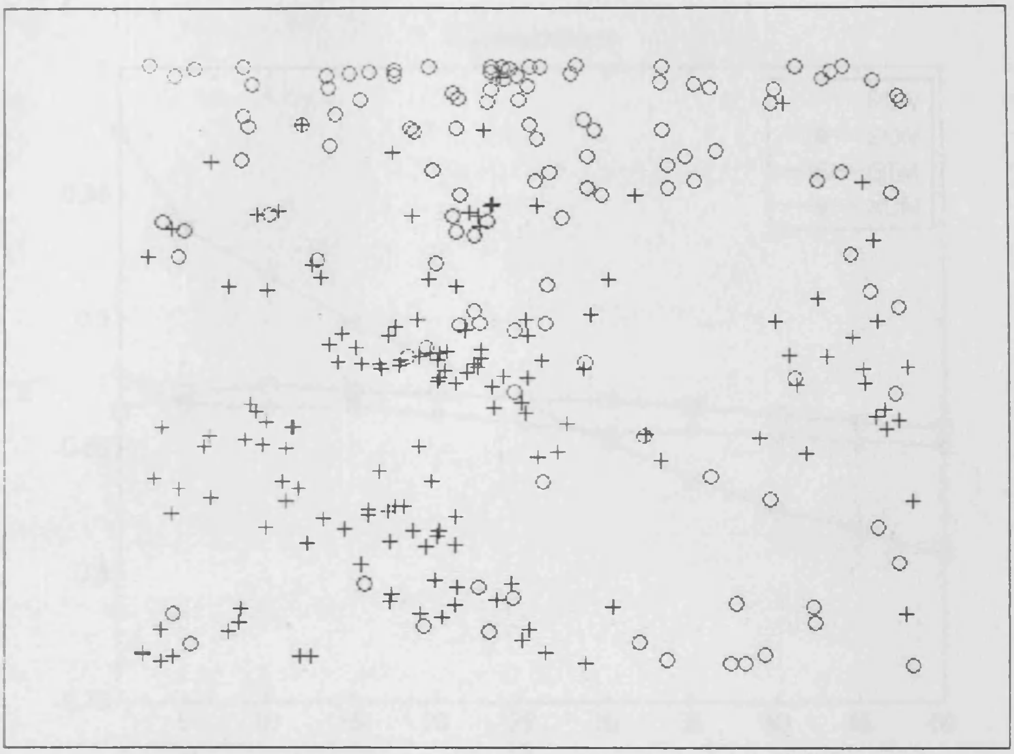

Figure 5.10(c) Heart data set projection using GTM

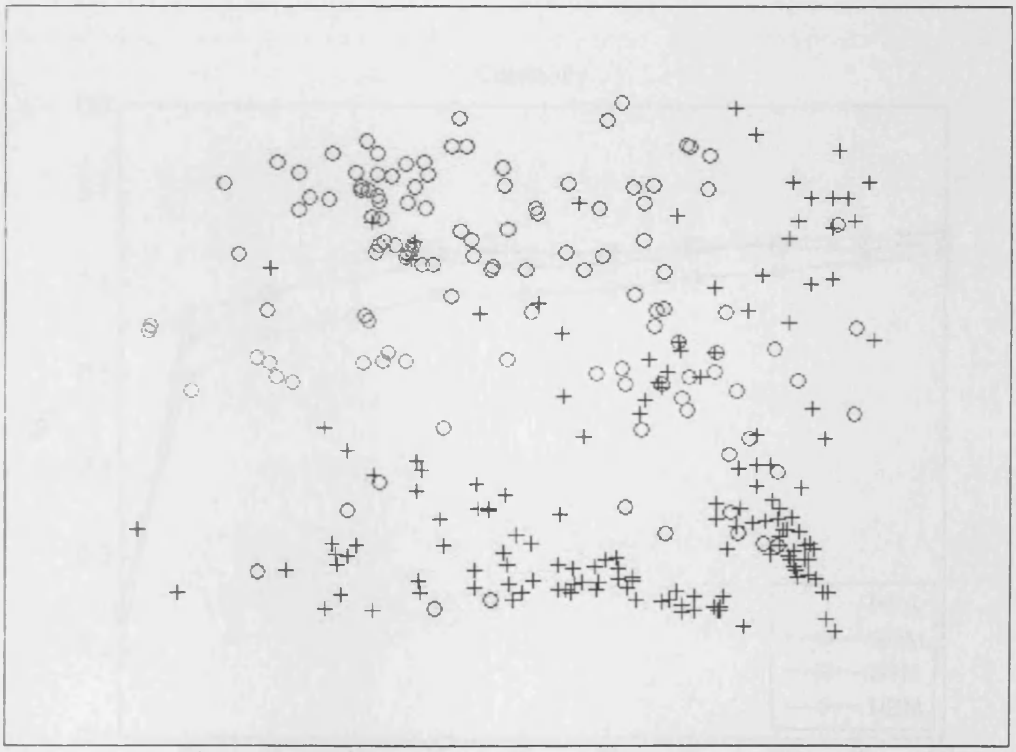

Figure 5.10(d) Heart data set projection using NBM

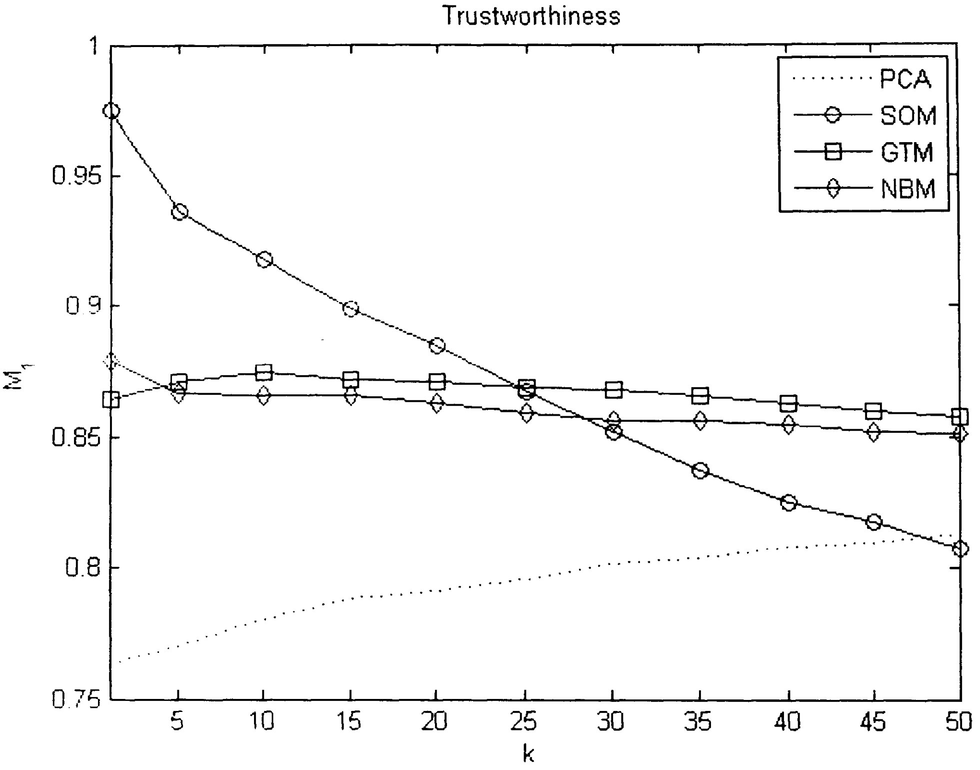

Figure 5.10(e) Trustworthiness measure for the Heart data set projection

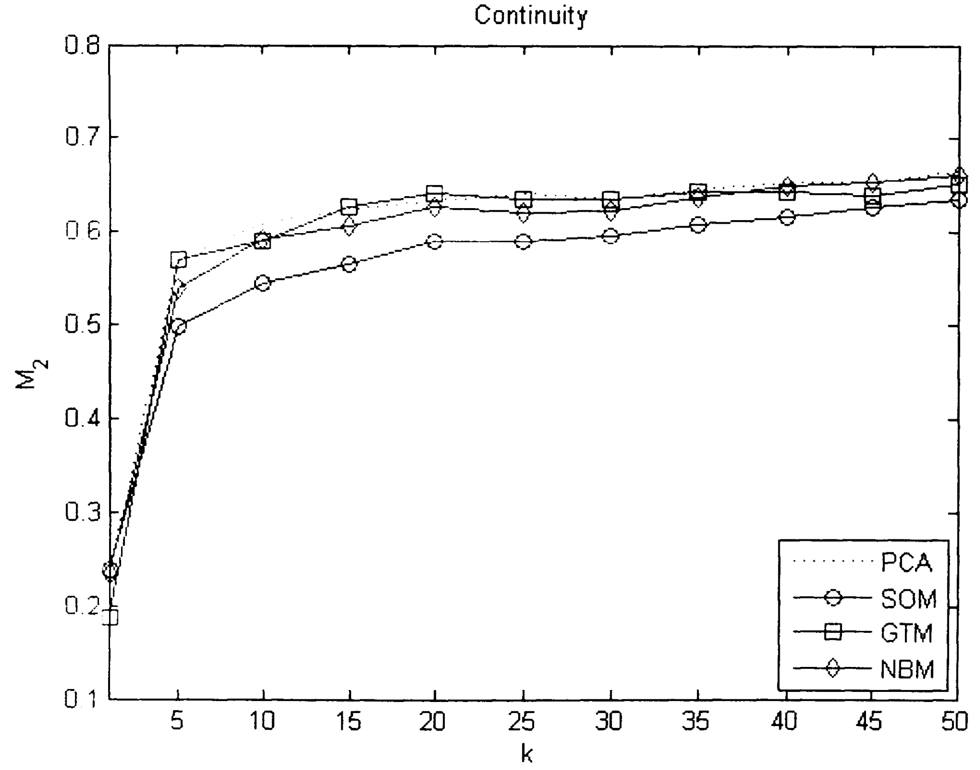

Figure 5.10(f) Continuity measure for the Heart data set projection

The Ionosphere data set (Fig. 5.11(a) – Fig. 5.11(f)). The PCA projection (Fig. 5.11(a)) shows a semi-dense cloud of points, the "O" class is mostly concentrated in the middle of the map, and the "+" class on the right and left side. Significant overlapping between the two classes can be appreciated. The SOM projection (Fig. 5.11(b)), shows the distribution of the two classes relatively clear, the "+" class is located on the left and right side of the map, while the "O" class is on the top and bottom part of the map. A few neurons containing points from both classes can be seen. The GTM projection (Fig. 5.11(c)), presents a sparse mapping concentrating the "+" class mostly in the upper right side of the map while the "O" class is surrounding the outside of "+" class. Some overlapping can be seen in the "+" class region with some "O" class points. The NBM projection (Fig. 5.11(d)), also has a sparse projection concentrating the "O" class in the centre and lower parts of the map, and the "+" class on the left side, upper part, and right side of the map. Some overlapping on the upper part and right side of the map can be seen. Analysing the Trustworthiness measure M_1 (Fig. 5.11(e)), the PCA projection obtains the lowest values with $0.81<M_1<0.84$ for all k computed. The SOM achieves the best trustworthiness for k up to 5, then, it drops significantly. The NBM starts at a similar value as GTM but then drops following the same behaviour as the SOM. The GTM for $k = 10$ and onwards achieves the best results. For the Continuity measure M_2 (Fig. 5.11(f)), the NBM performs significantly better than PCA, GTM, and the SOM projections. These last three methods show similar values for all k.

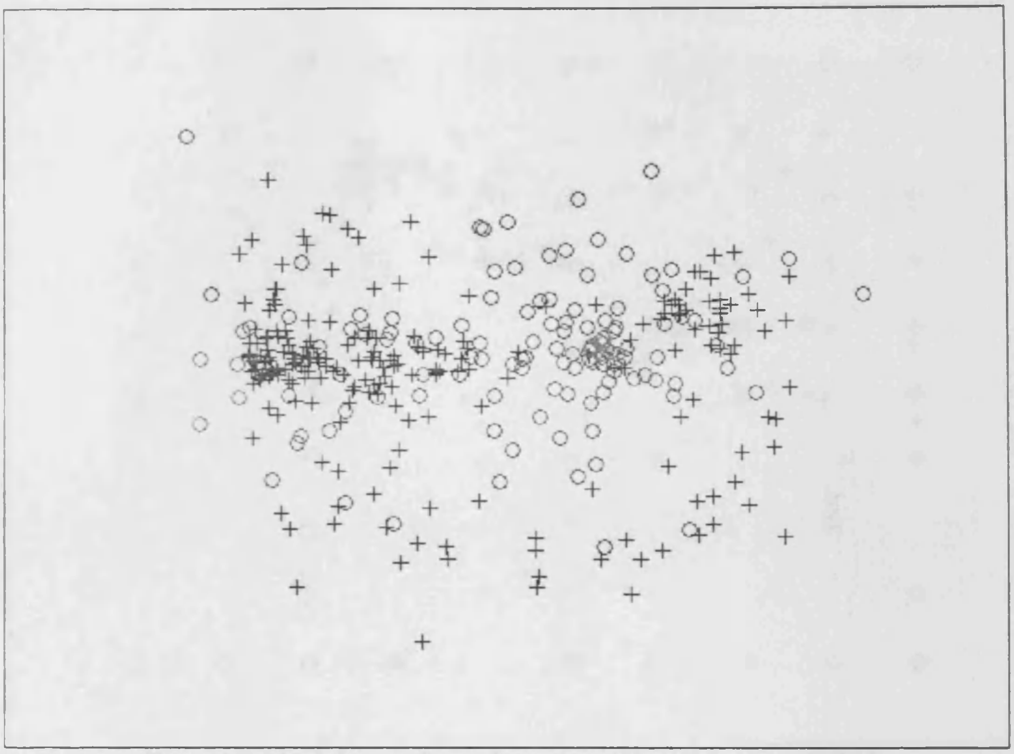

Figure 5.11(a) Ionosphere data set projection using PCA

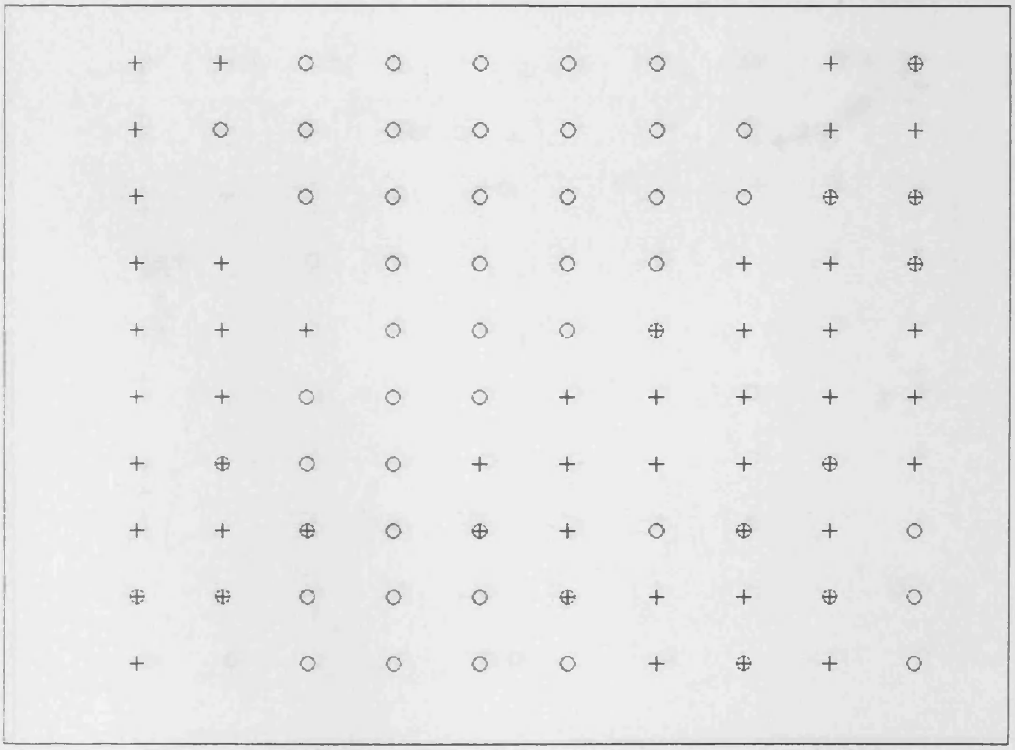

Figure 5.11(b) Ionosphere data set projection using SOM

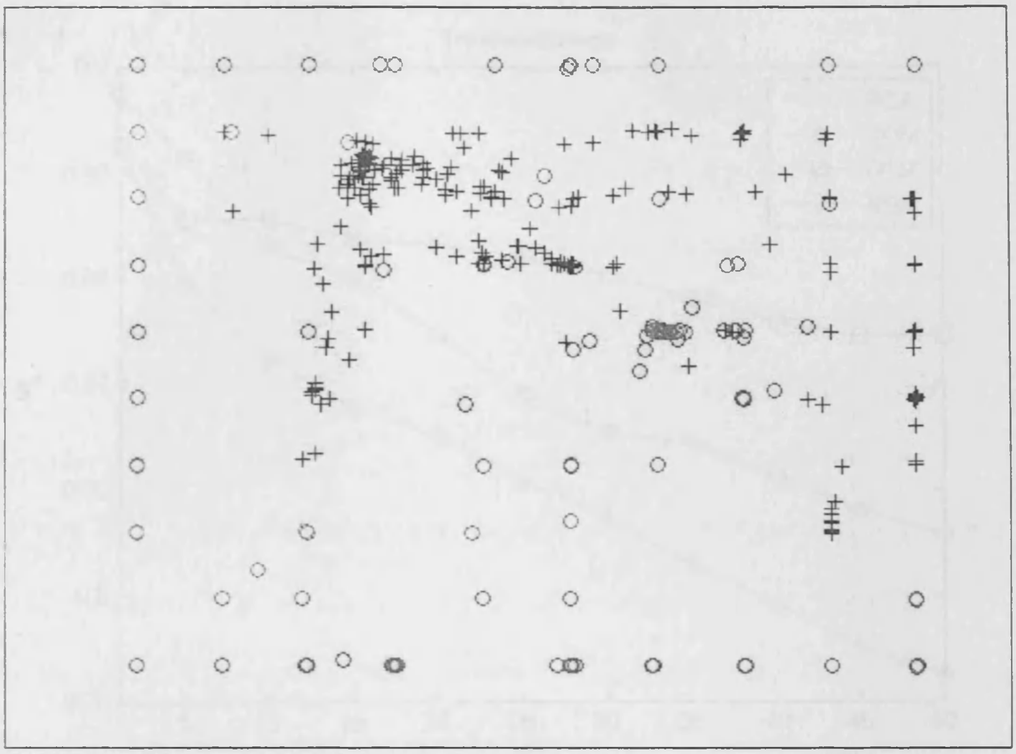

Figure 5.11(c) Ionosphere data set projection using GTM

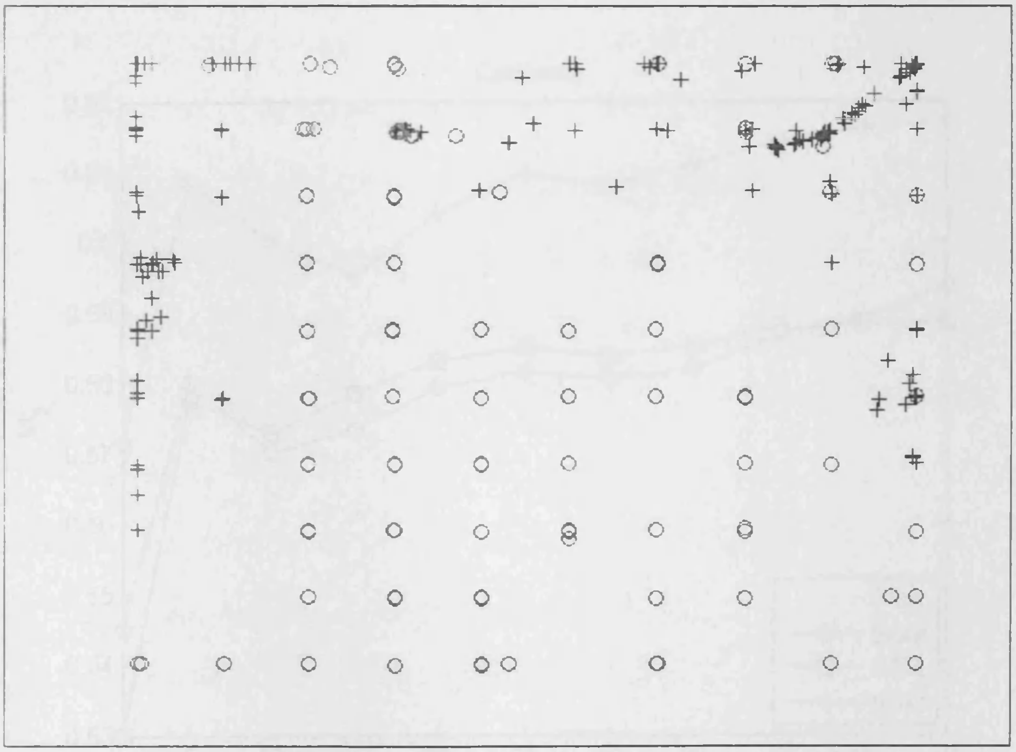

Figure 5.11(d) Ionosphere data set projection using NBM

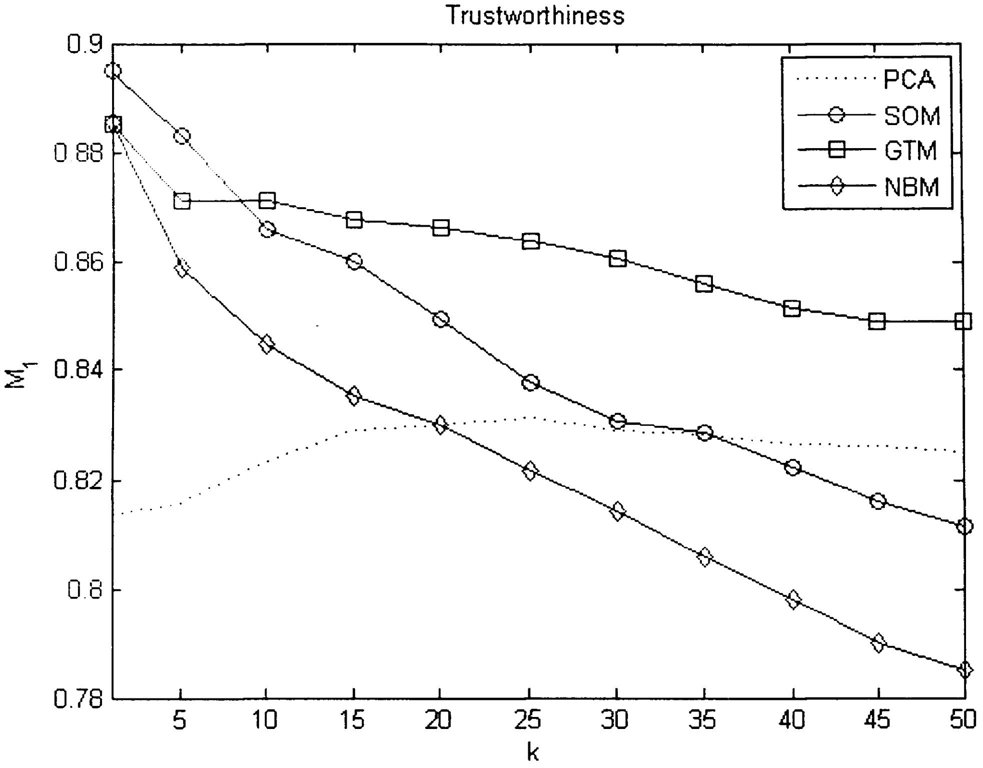

Figure 5.11(e) Trustworthiness measure for the Ionosphere data set projection

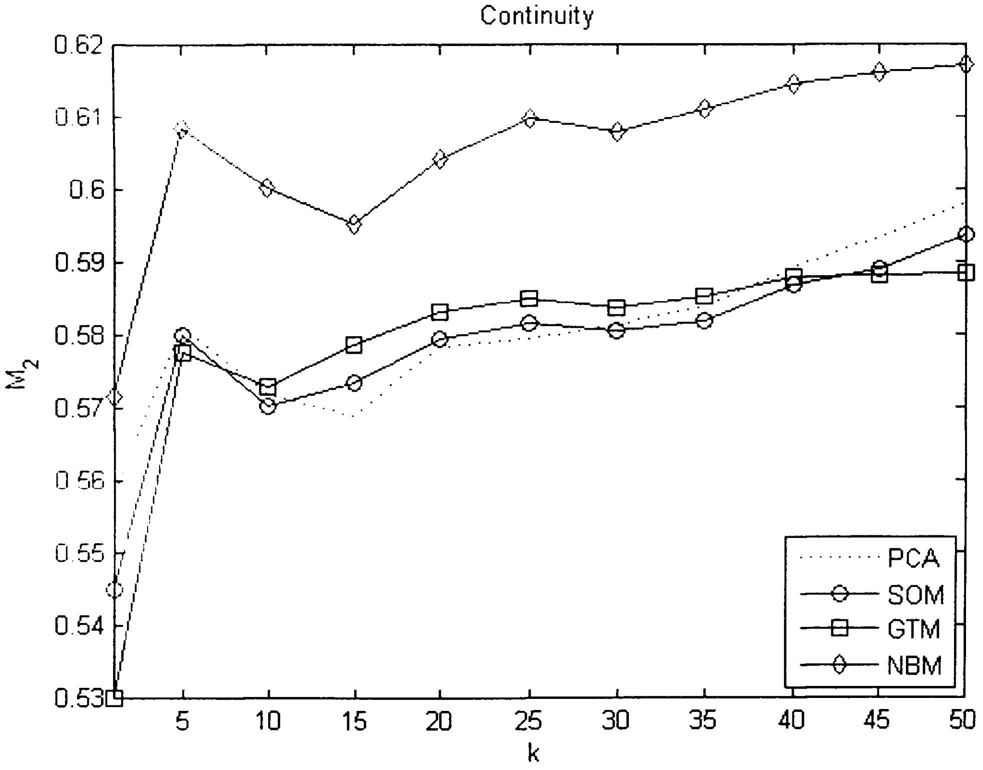

Figure 5.11(f) Continuity measure for the Ionosphere data set projection

The Iris data set (Fig. 5.12(a) – Fig. 5.12(f)). The PCA projection (Fig. 5.12(a)) shows two clouds of points, one which only contains points from the "*" class, and then the other cloud which has the "O" class on the lower left side and the "+" class on the upper right side. Points from the "*" class are clearly linearly separable from the class "O" and "+". The class "+" and "O" are non-linearly separable which is shown by the slight overlapping of these two classes where they intersect. The SOM projection (Fig. 5.12(b)), shows the distribution of the three classes very clear. The "*" class is located in the top right region of the map, the "+" class is located in the middle region of the map (in a diagonal distribution) and the "O" class is in the bottom left part of the map. Only three neurons contain points from the "+" and "O" class. The GTM projection (Fig. 5.12(c)), presents a sparse mapping concentrating the "*" class on the top left and left side of the map, while the other two classes are located on the lower right side of the map, very little overlapping can be seen. The NBM projection (Fig. 5.12(d)), is quite similar to the PCA mapping in this case, although the "*" class, located on the left side of the map, is more dense in the NBM. Points from the "+" and "O" class are on the right side of the map, where little overlapping can be seen in the regions where they intersect. Analysing the Trustworthiness measure M_1 (Fig. 5.12(e)), the PCA projection obtains the best values with $M_1>0.97$ for all k computed. Then it is followed closely by the NBM. The GTM is similar to NBM at k =5, then, it drops ending at a value close to 0.92 for k =50. The SOM, performing slightly worst than GTM, follows the same drop as GTM and ends at 0.91. In general the 4 projections obtain values $M_1>0.91$ for all k, which reveals the good quality of the projections for this data set. For the Continuity measure M_2 (Fig. 5.12(f)), the NBM performs the best and similar to the PCA. The GTM and the SOM projections present similar values for all k.

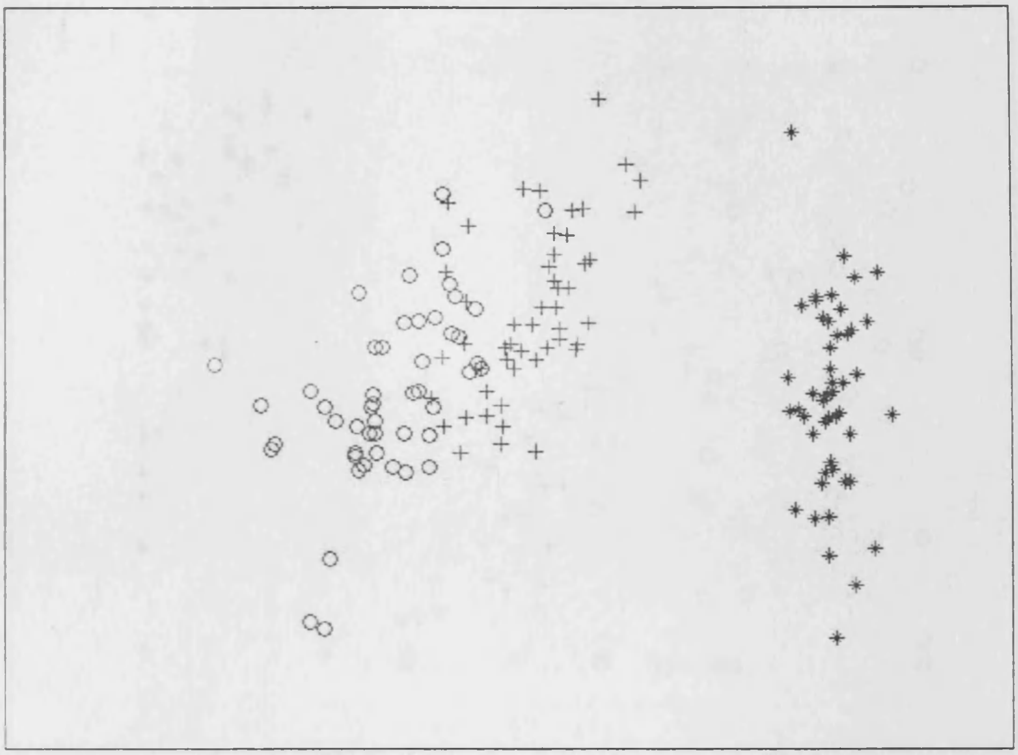

Figure 5.12(a) Iris data set projection using PCA

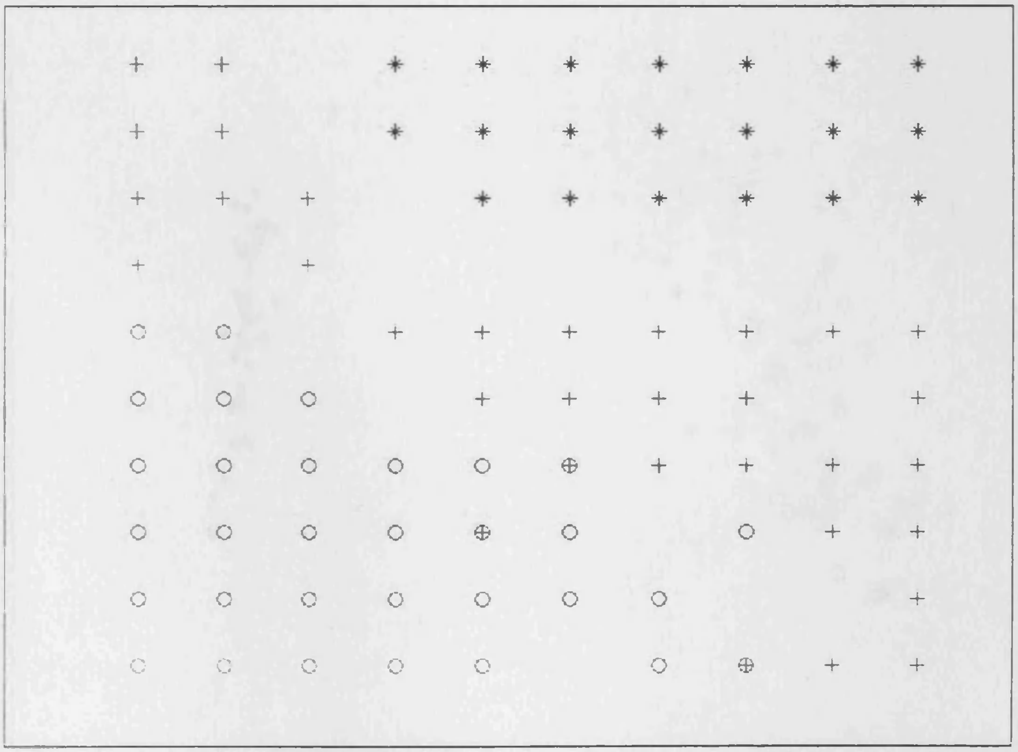

Figure 5.12(b) Iris data set projection using SOM

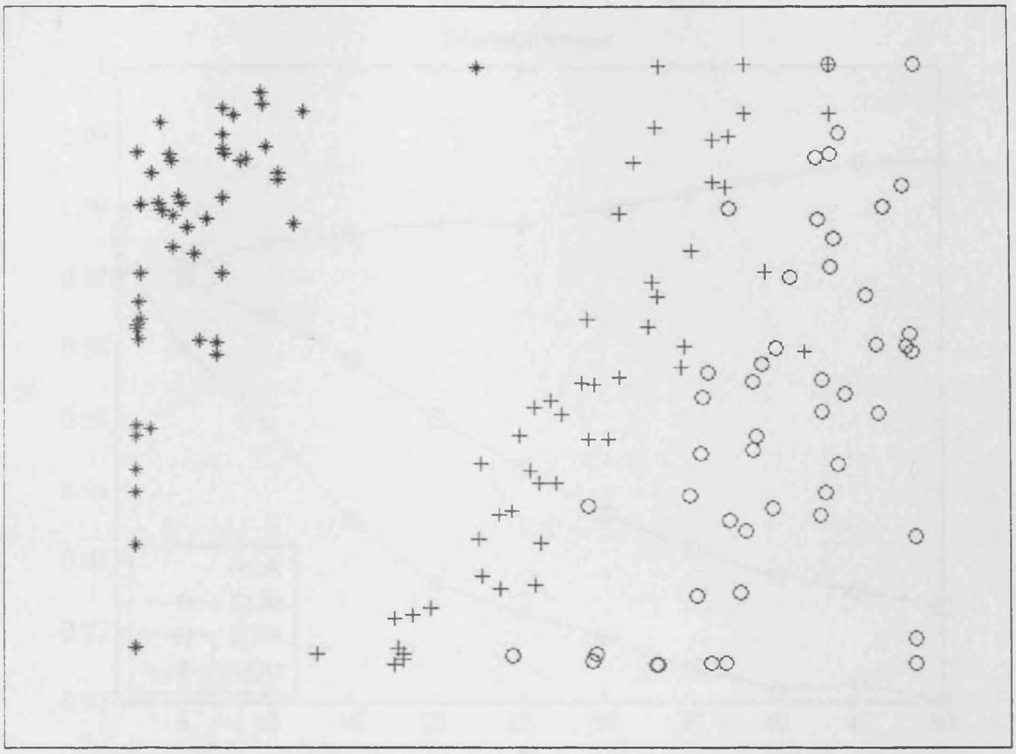

Figure 5.12(c) Iris data set projection using GTM

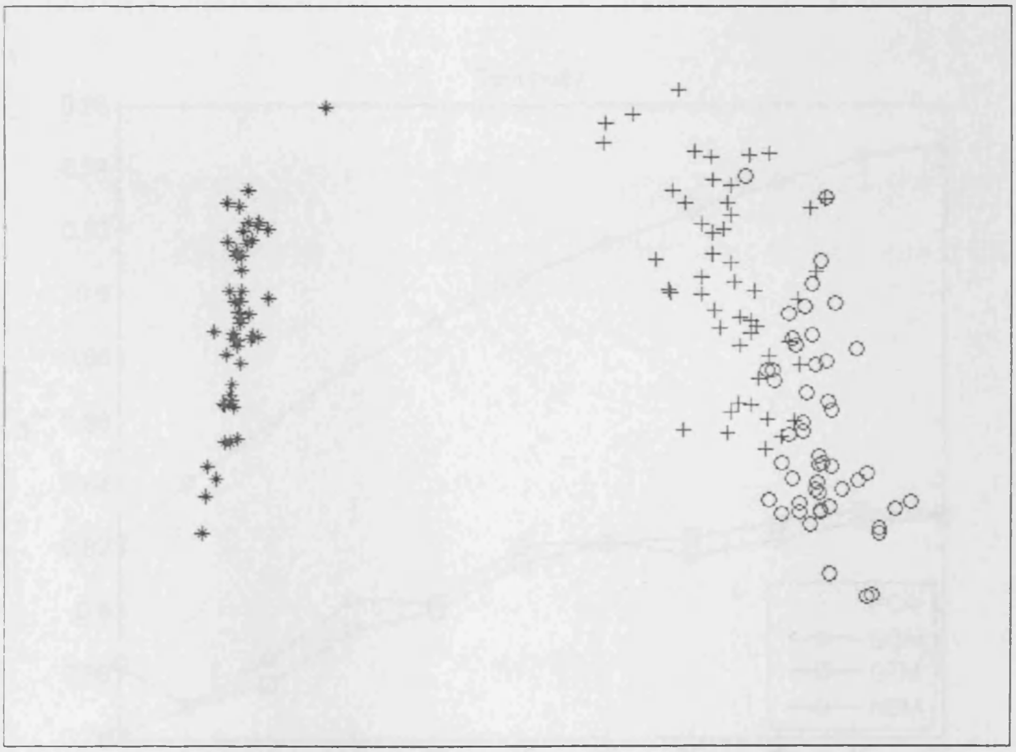

Figure 5.12(d) Iris data set projection using NBM

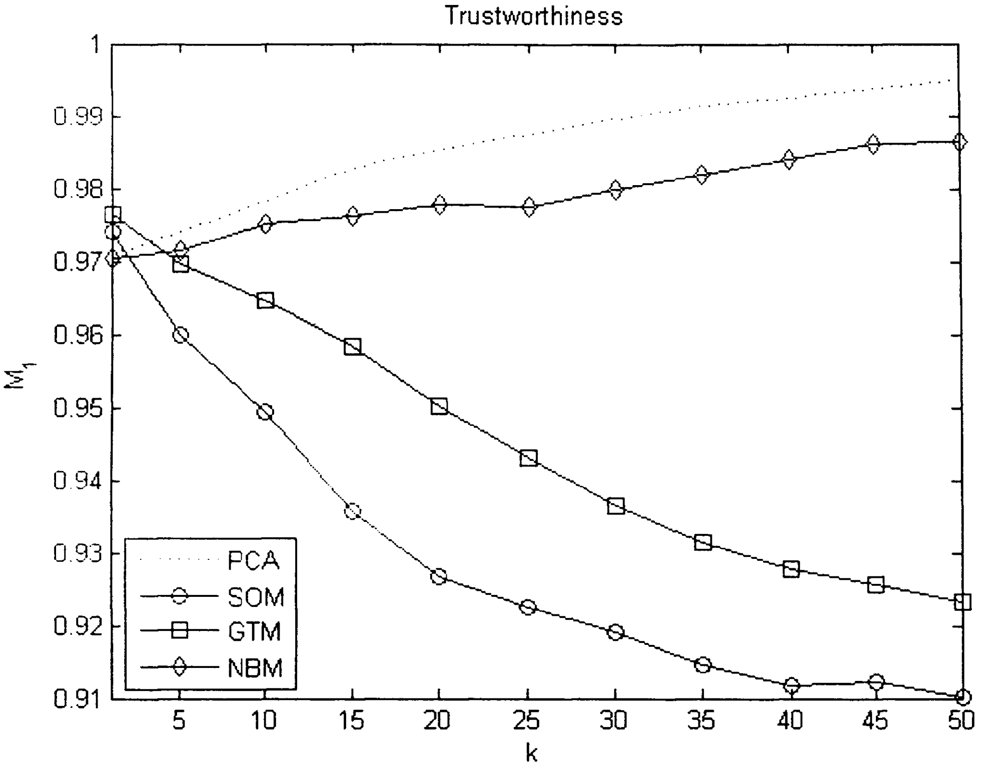

Figure 5.12(e) Trustworthiness measure for the Iris data set projection

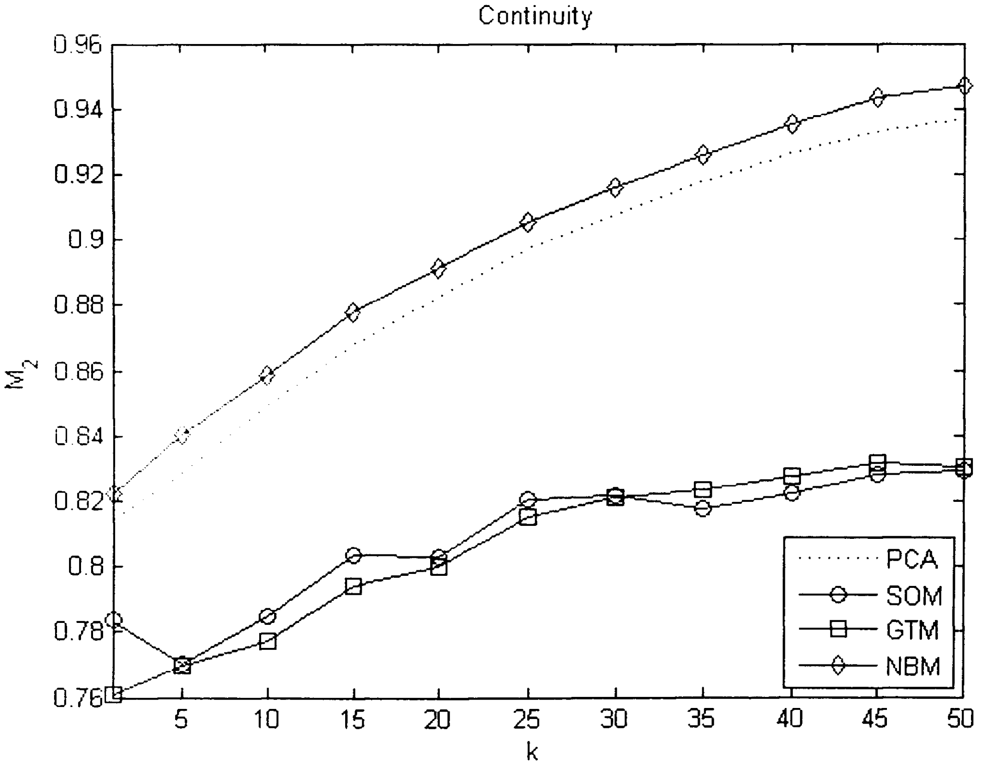

Figure 5.12(f) Continuity measure for the Iris data set projection

The Pima data set (Fig. 5.13(a) – Fig. 5.13(f)). The PCA projection (Fig. 5.13(a)) shows a semi-dense cloud of points, the "O" class is mostly concentrated on lower and left side of the mapping, and the "+" class on the top right side, significant overlapping can be appreciated as well. The SOM projection (Fig. 5.13(b)), shows most of its neurons containing points from both classes, only the "+" class has a distinguishable region in the map while the "O" class has only four isolated neurons. The GTM projection (Fig. 5.13(c)), presents a sparse mapping concentrating the "+" class mostly in the centre, left and top left of the map and the "O" class in the bottom and right side of the map. Significant overlapping can be viewed as well. The NBM projection (Fig. 5.13(d)), also has a sparse projection concentrating the "O" class in the lower right region of the map, and the "+" class on the upper left region. High overlapping regions are also present. Analysing the Trustworthiness measure M_1 (Fig. 5.13(e)), the PCA projection obtains the lowest values with $0.8<M_1<0.84$ for all k computed, which is not bad. The SOM achieves the best trustworthiness for all k. The NBM outperforms the GTM for all k. For the Continuity measure M_2 (Fig. 5.13(f)), the NBM projection obtains the best result for all k, and then it follows the PCA, the GTM, and finally the SOM.

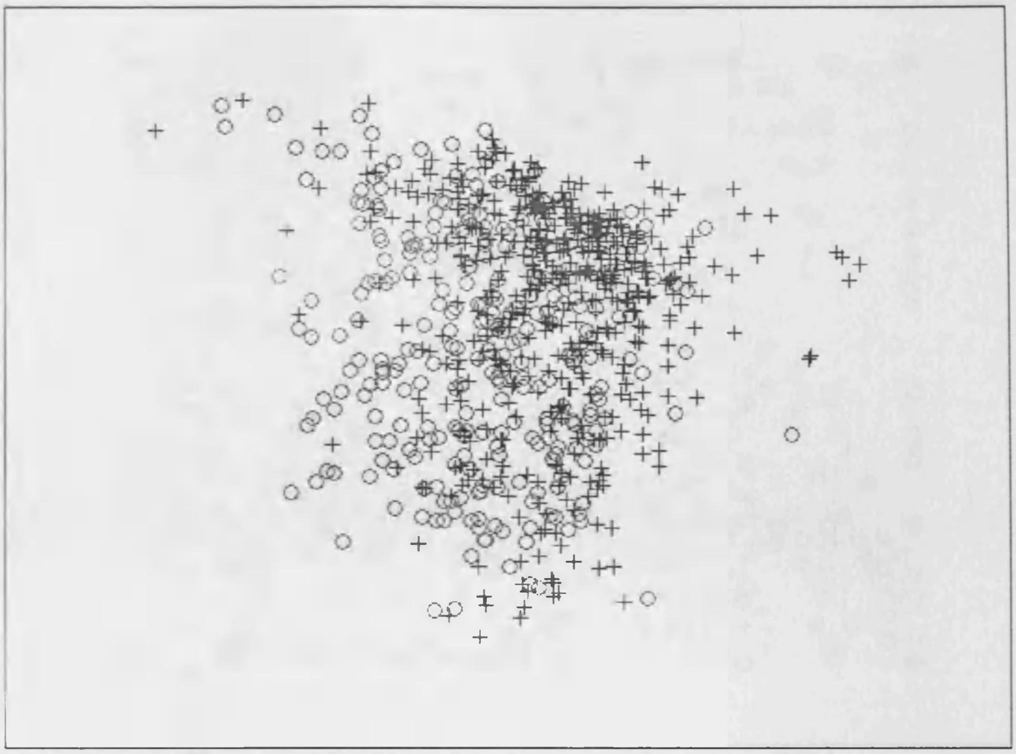
Figure 5.13(a) Pima data set projection using PCA

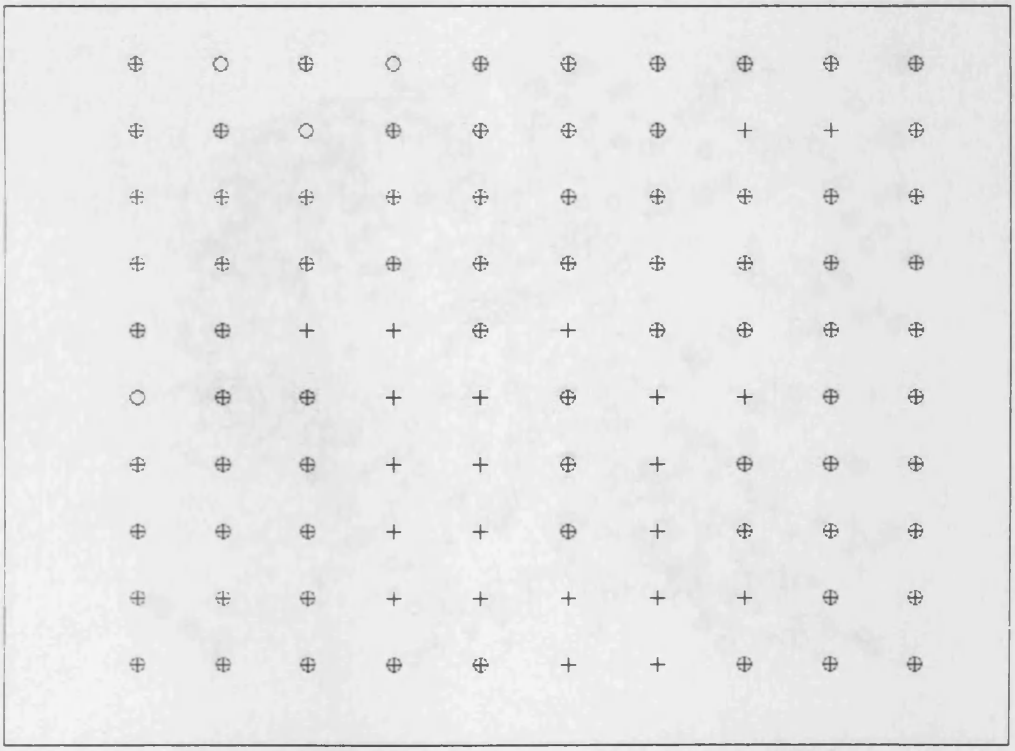
Figure 5.13(b) Pima data set projection using SOM

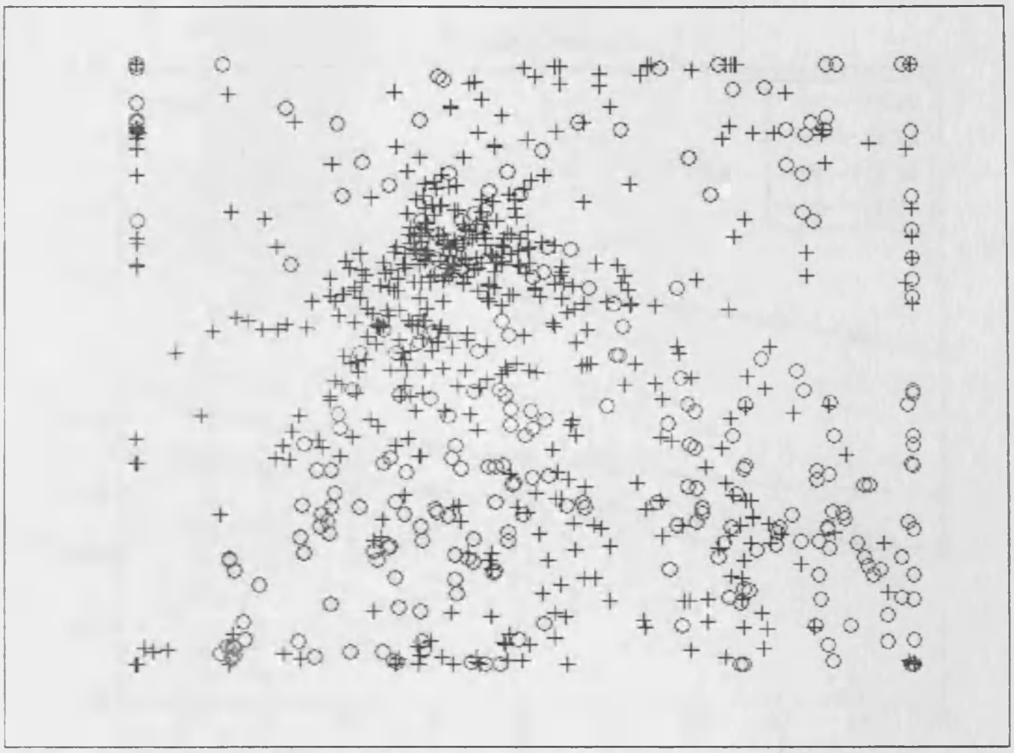

Figure 5.13(c) Pima data set projection using GTM

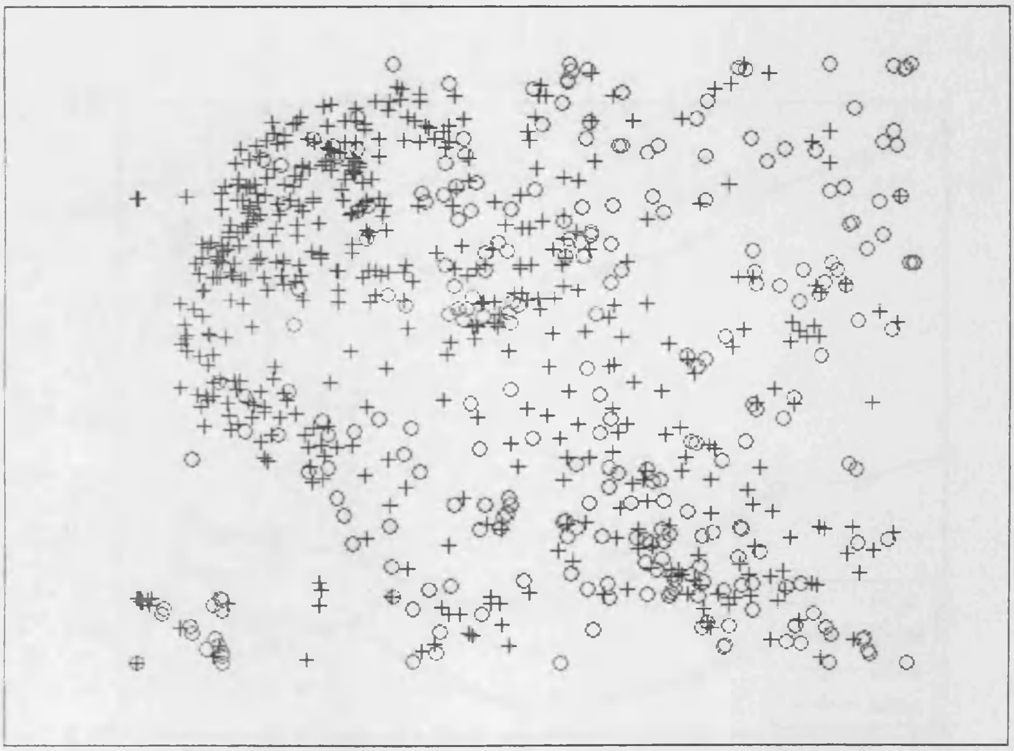

Figure 5.13(d) Pima data set projection using NBM

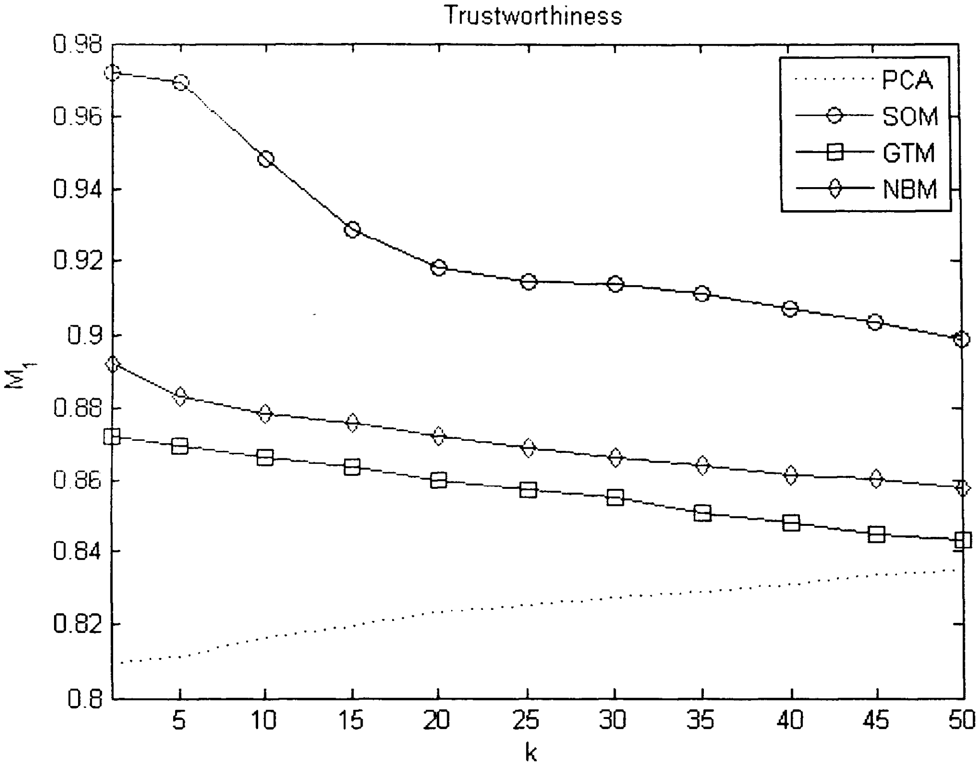

Figure 5.13(e) Trustworthiness measure for the Pima data set projection

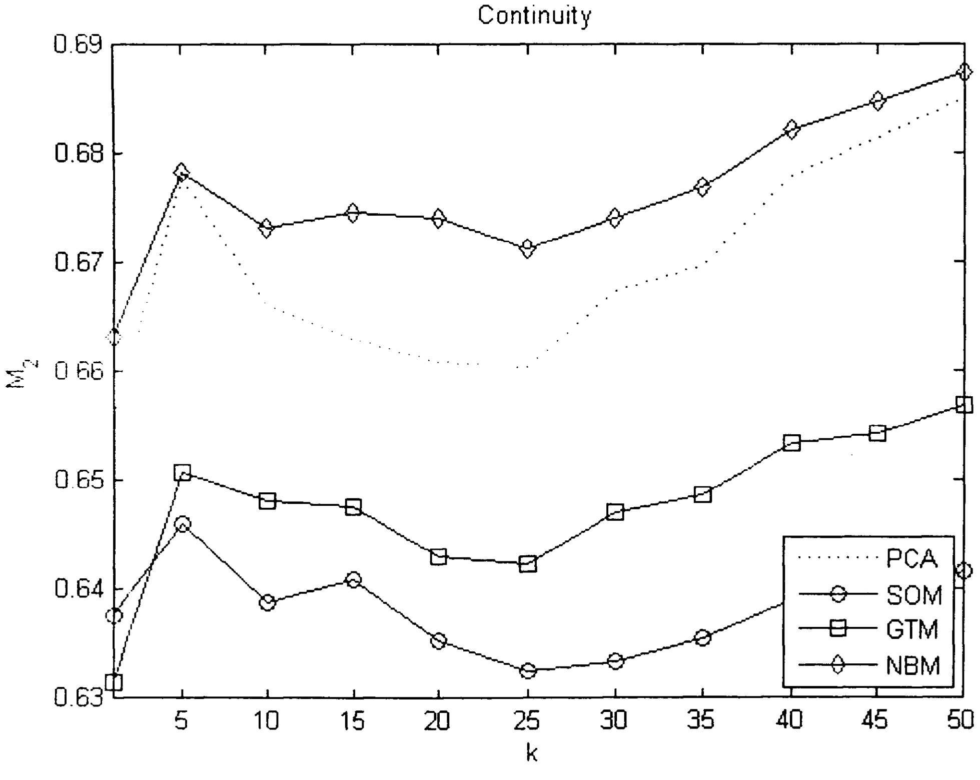

Figure 5.13(f) Continuity measure for the Pima data set projection

The Wine data set (Fig. 5.14(a) – Fig. 5.14(f)). The PCA projection (Fig. 5.14(a)) shows three clouds of points, one which only contains points from the "*" class, located in the lower left side of the projection, another cloud which has the "+" class on the upper centre and the "O" class on the lower right side of the map. The SOM projection (Fig. 5.14(b)), shows the distribution of the three classes very clear. The "*" class is located in the right region of the map, the "+" class is located in the top, middle, and lower regions of the map and the "O" class is in the bottom left part of the map. Only two neurons contain points from the "+" and "O" class and one neuron contains points from the "+" and "*" class. The GTM projection (Fig. 5.14(c)), presents a sparse mapping concentrating the "*" class on the top right, top centre of the map, while the "O" class is located in the bottom right and bottom centre of the map, the "+" class is mostly concentrated on the left side of the mapping, although a few points from this class can be found in other parts of the map. The NBM projection (Fig. 5.14(d)), shows a sparse projection as well (when compared to the PCA) where the "*" class is located on the bottom right side of the map, while the "O" is on the opposite upper left side. The "+" class is on the top right side, and top middle of the map. Like the GTM, a few points from this class can be found in other parts of the map. Analysing the Trustworthiness measure M_1 (Fig. 5.14(e)), the SOM projection obtains the best values for k up to 10, then, it is outperformed by the GTM and NBM. The GTM and NBM show similar results for k =5, after that, the GTM shows slightly better values up to k =35 then it is passed by the NBM. The PCA obtains the lowest initial values but then outperforms the SOM from k =25 onwards. In general GTM and NBM have M_1>0.92 values for all k. For the Continuity measure M_2 (Fig. 5.14(f)), the PCA obtains the highest value for k =5, then the PCA, GTM, and the NBM show similar values for the rest of the k values. The SOM projection has the lowest values for all k, with the minimum value at $k = 5$.

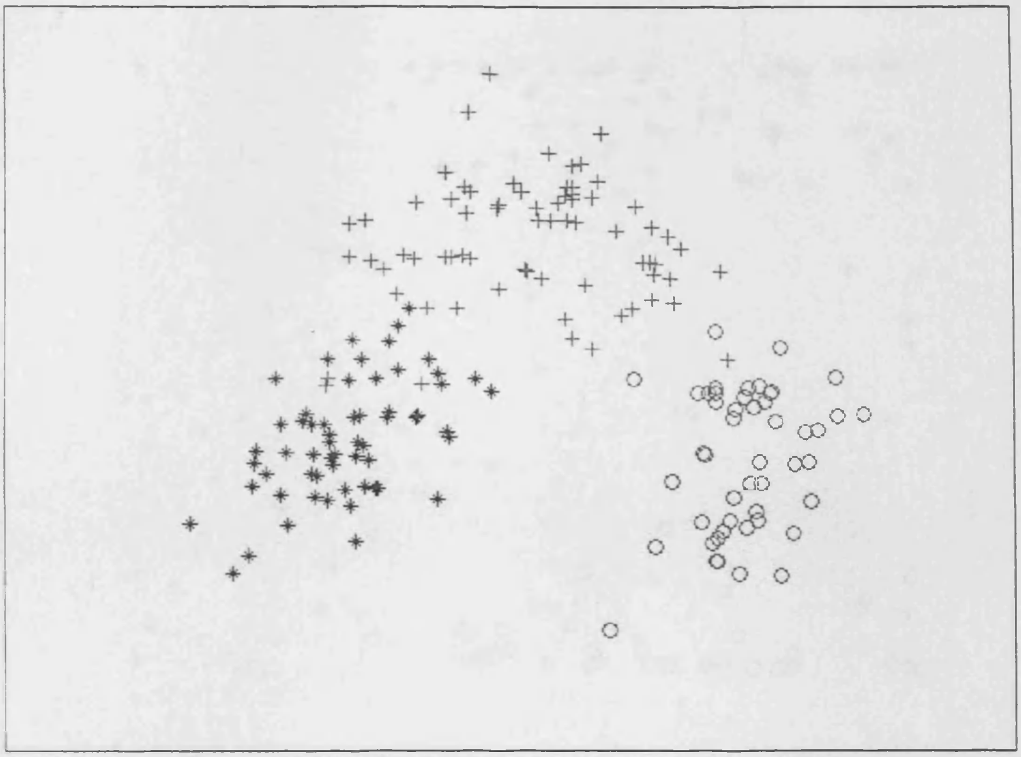

Figure 5.14(a) Wine data set projection using PCA

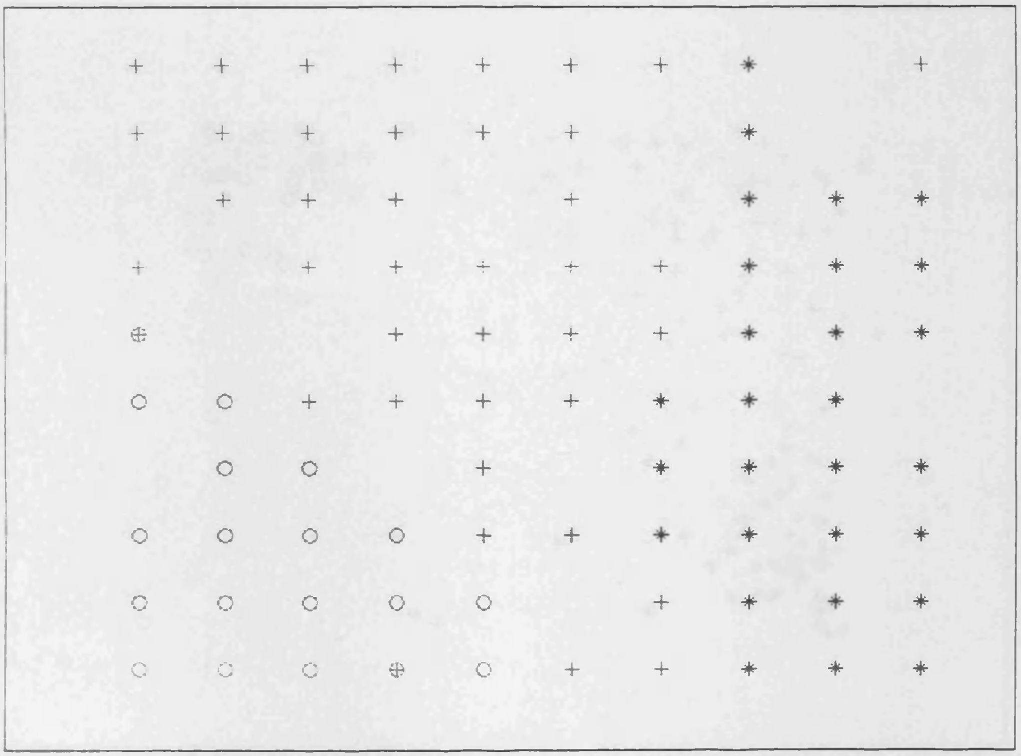

Figure 5.14(b) Wine data set projection using SOM

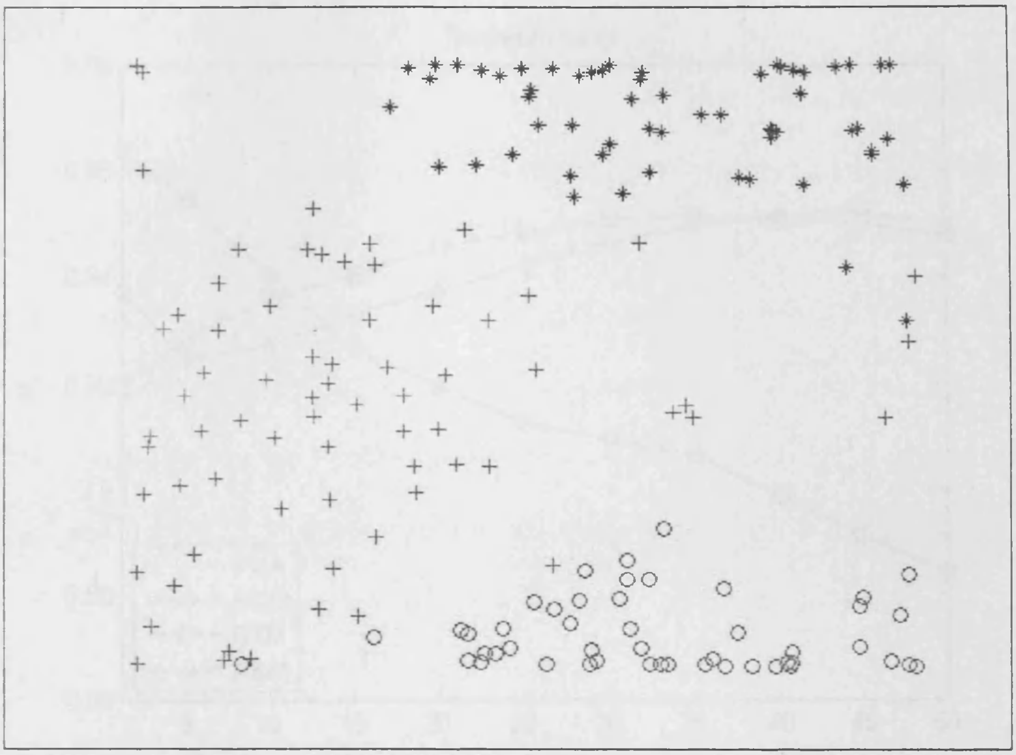

Figure 5.14(c) Wine data set projection using GTM

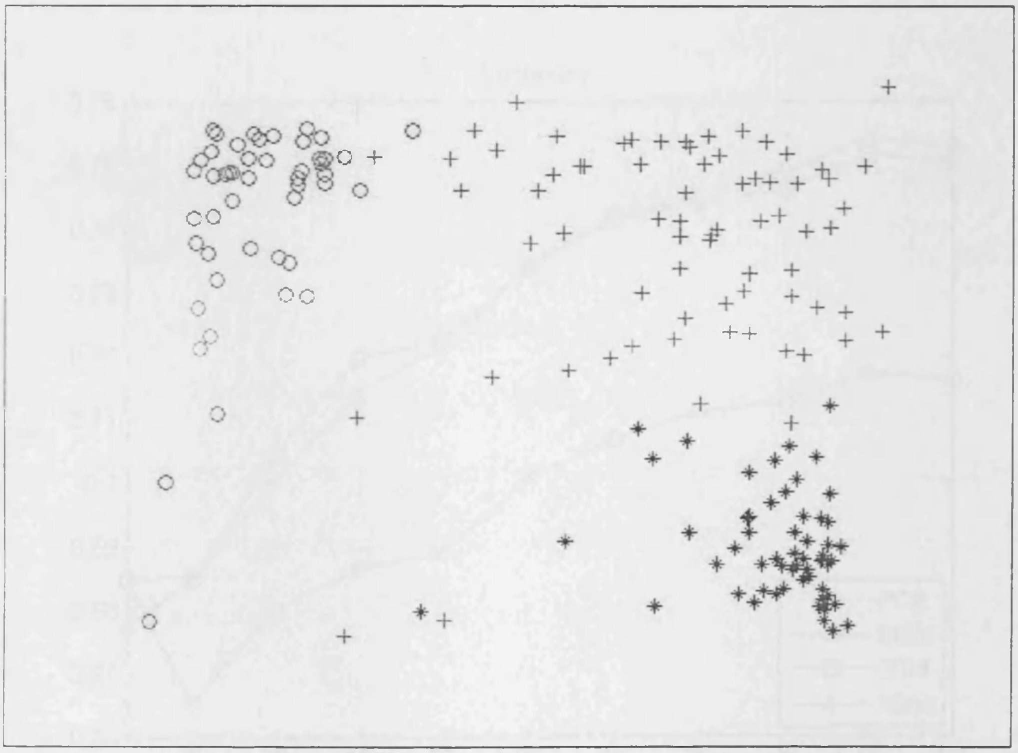

Figure 5.14(d) Wine data set projection using NBM

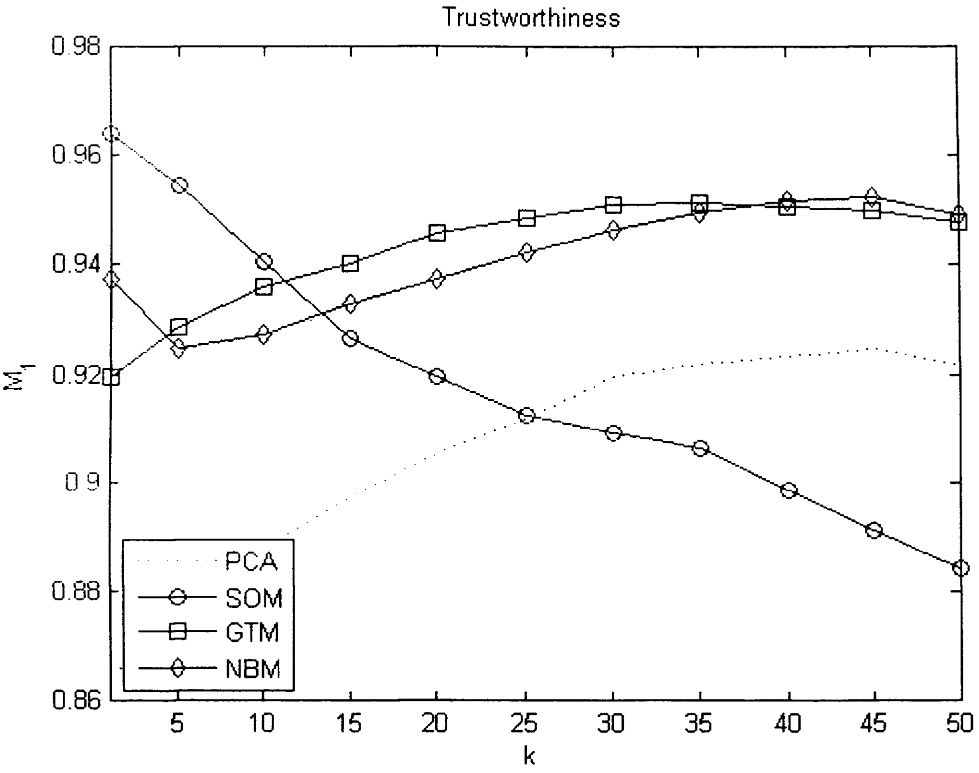

Figure 5.14(e) Trustworthiness measure for the Wine data set projection

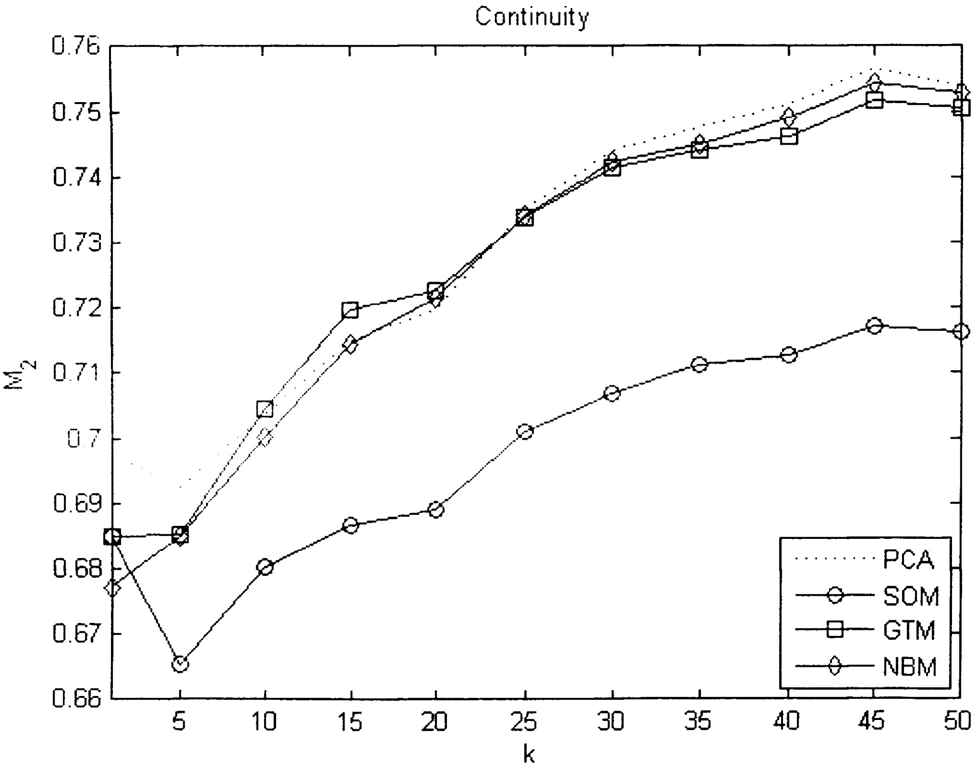

Figure 5.14(f) Continuity measure for the Wine data set projection

In general, the results using the ten benchmark data set, shows that the SOM obtains the best trustworthiness for small values of *k* (*k*<15), it is followed closely by the GTM and NBM which present a similar behaviour, noticing that for small values of *k* the NBM would obtain slight better results than GTM. For larger values of *k*, in several cases the GTM and NBM obtained better results than the SOM. The PCA had the lowest values, except when the data set was clearly linearly separable like the iris and wine data set. For the Continuity measure the NBM and the PCA performed the best whilst the SOM obtain the worst over all performance. This can give an insight of the trade off between trustworthiness and continuity, since although the SOM performed very well in M_1 for small values of *k*, it does this by compromising its performance in M_2, on the other hand the NBM and GTM obtained results which are much more balanced for both measures. Analysing the mappings visually, the nonlinear projection methods clearly outperforms the PCA, which only has nice visualizations (non dense) for simple data sets with little overlapping. The GTM and NBM obtained the best results due to its more sparse projection which enables better separation amongst classes. The SOM, due to its discrete output, although it also achieves good separation in some data sets it is very difficult to appreciate how the data is distributed if no label where used. Also it is important to point out that out of the ten simulations for each data set, in the case of the SOM, the map selection process was done visually thanks to the class labels, where the map with best class separation was selected. While the GTM and NBM were selected automatically using the log-likelihood measure only.

5.4.4 Industrial application

This subsection describes the application of the proposed algorithm to the visualisation of the wood data. This data was generated from a low-cost Automatic

Visual Inspection (AVI) system for wood defect detection (Estévez *et al.*,2003). The feature extraction module from the AVI system extracted features from objects and windows of 64 by 64 pixels centred in the object geometrical centre. The features used in this work include: 7 object geometrical features measured on the binarised grey image (e.g.: area, perimeter, average radius, aspect ratio, etc.); 96 object colour features (24 features measured in each of the four channels, R, G, B, and grey); 46 window colour features (e.g., mean and variance of window histograms, mean and variance at the edge of windows); and 16 co-occurrence features (which contain texture information) which were added recently in Ruz *et al.* (2008). In total there are 165 features computed from the segmented defects. The data set used in this work consists in 2247 examples (wood board images of 320 × 240 pixels) which have been manually labelled/classified into one of the following 10 defect categories (see Ruz *et al.* (2005) for details): birdseye, pockets, wane, split, stain, blue stain, pith, dead knot, live knot, and hole. Also, the clear wood category was added, which corresponds to non-defective wood board images. In total there are 11 classes (labelled from 0 to 10 respectively), with approximately 200 examples per class.

For this simulation, the SOM, GTM, and the NBM used a 20 x 20 grid output. The GTM used 81 basis functions, the maximum number of epochs was set to 30 for the GTM and NBM, 500 for the SOM. For each technique, five repetitions were made, choosing the best map following the criteria explained in the previous experiments. The other parameters remained the same as before.

Results of these simulations are shown in Fig. 5.15(a) – Fig. 5.15(f). The PCA projection (Fig. 5.15(a)) shows a dense cloud of points where the eleven different classes highly overlap. Only class 3 and class 0, in the bottom part of the cloud of points, a more distinguishable. The other regions are much denser. The SOM projection (Fig. 5.15(b)) presents a much clearer separation amongst the classes, The GTM (Fig. 5.15(c)) presents a slightly more sparse projection but the separation of the classes is not as clear as the SOM. The NBM (Fig. 5.15(d)) achieves good separation amongst the different classes similar to SOM. Analysing the Trustworthiness measure M_1 (Fig. 5.15(e)), the SOM projection obtains the best values for *k* up to 5, then, it is outperformed by the GTM and NBM only slightly. The GTM and NBM show similar results from $k = 10$ onwards. The three methods presented values $M_1 > 0.97$ for *k* up to 30, which is very good. As expected the PCA projection obtained the lowest performance for this measure. For the Continuity measure M_2 (Fig. 5.15(f)), the PCA obtained the highest values for all *k*. At $k = 5$ the NBM has it highest value and outperforms the SOM and GTM, then for values of *k* up to 25, the NBM and GTM obtain similar values, outperforming both the SOM.

An interesting conclusion was found when the projections were analysed in detail. Ruz *et al.* (2005) conducted a histogram-based study of the colour intensities from defective regions and grain line (clear wood) regions of the radiata pine board images. This is one of the main difficulties found in the image segmentation step of the AVI system because the clear wood (free from defects) boards contain dark grain lines. The study showed that two of the defect categories which overlapped the most with the clear wood (label 10) were the stain (label 4) and the blue stain (label 5) categories. Now, this conclusion can also be obtained by observing the maps projected by the SOM, GTM, and the NBM, where most of the overlapping is due to 10 overlapping with 5 or 10 overlapping with 4. This result shows, in part, the effectiveness of these projection techniques.

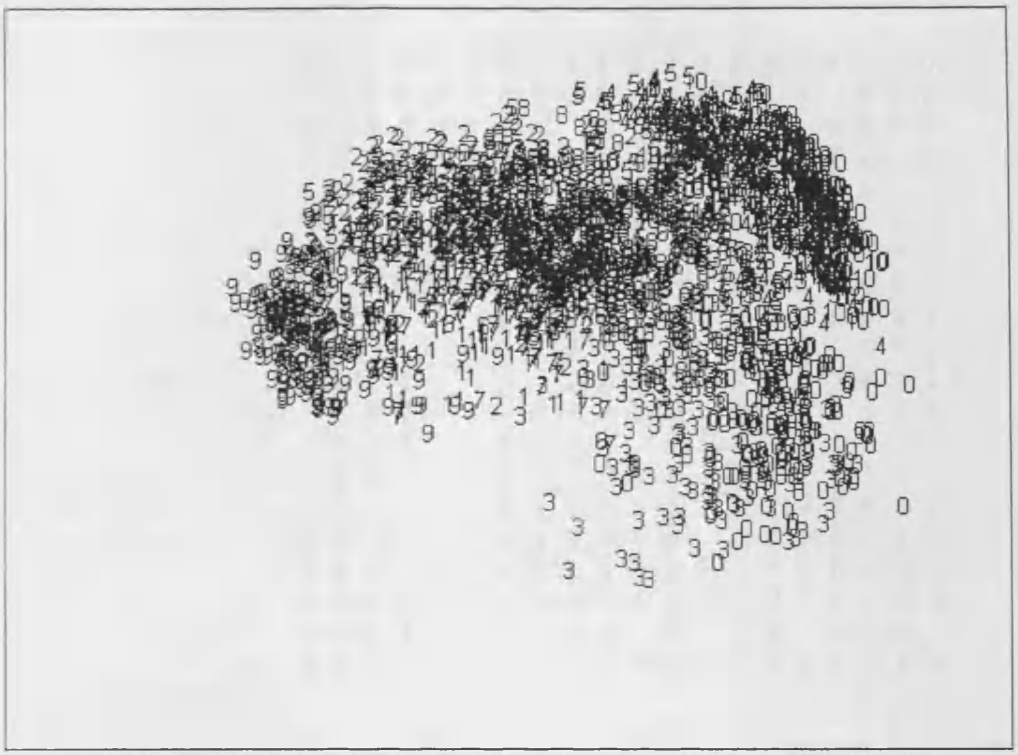

Figure 5.15(a) Wood data set projection using PCA

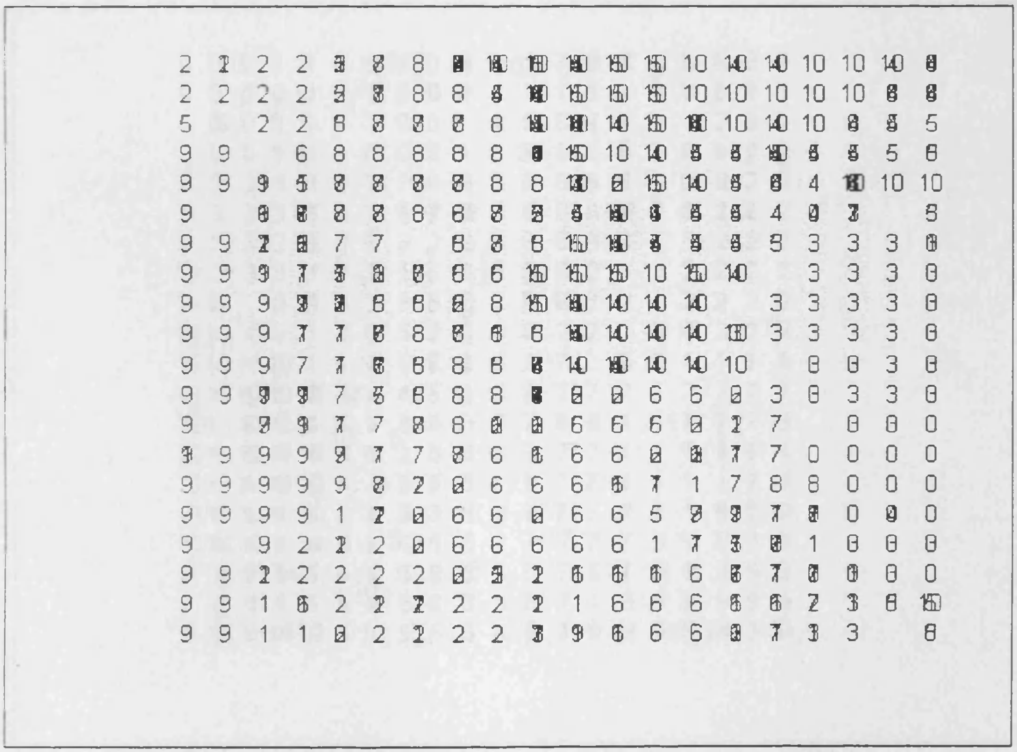

Figure 5.15(b) Wood data set projection using SOM

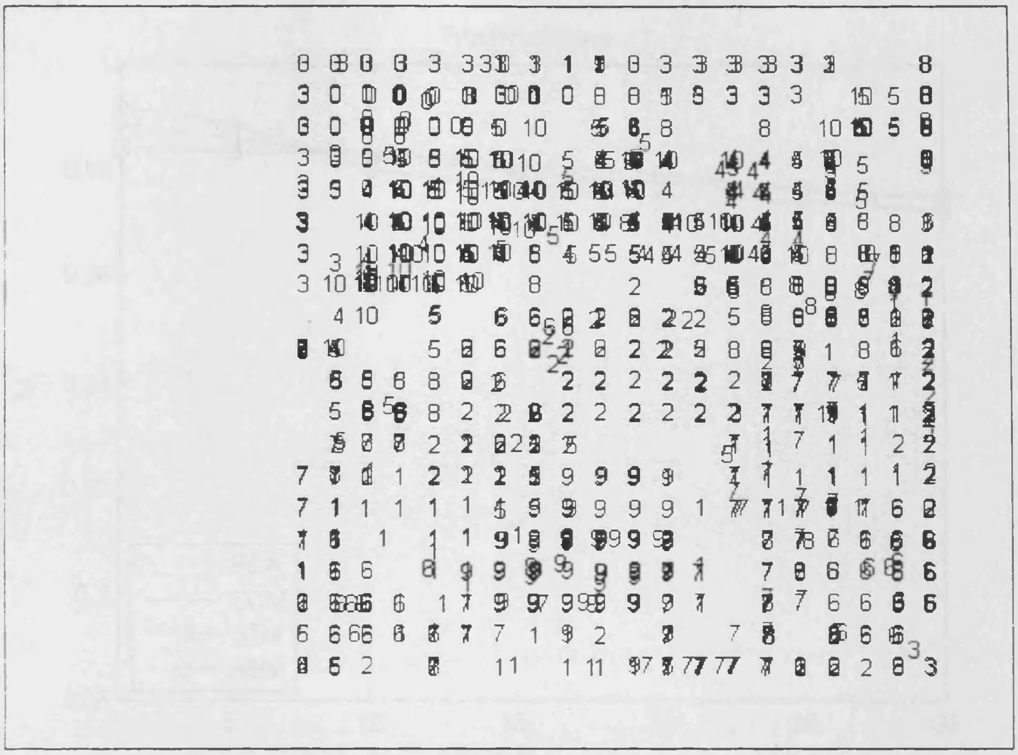

Figure 5.15(c) Wood data set projection using GTM

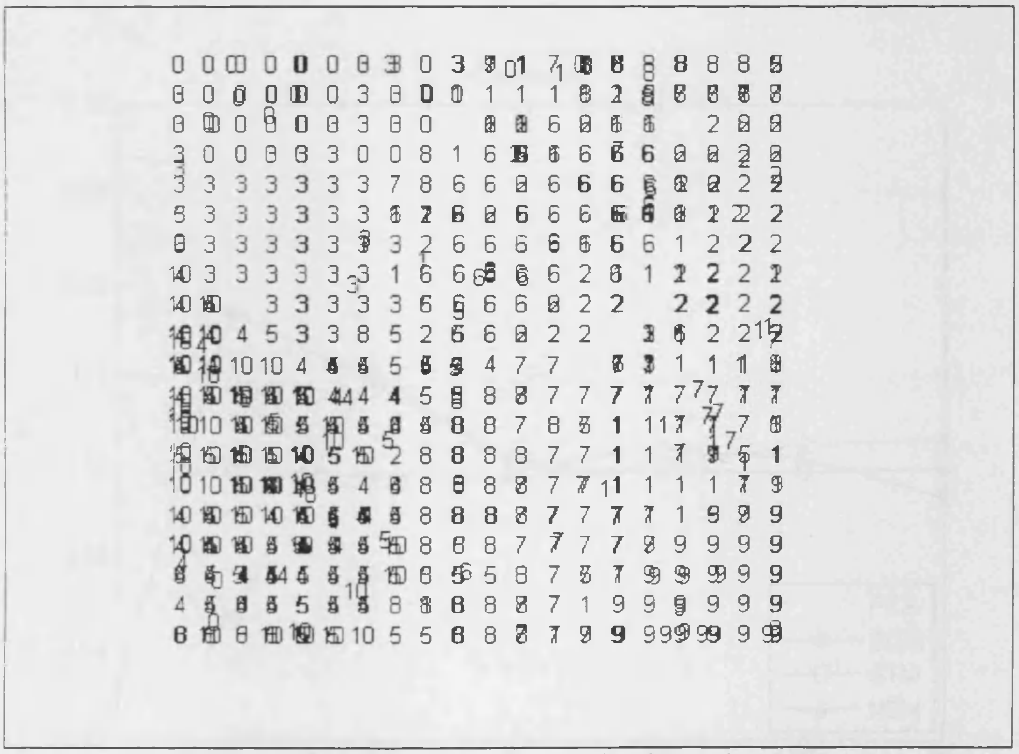

Figure 5.15(d) Wood data set projection using NBM

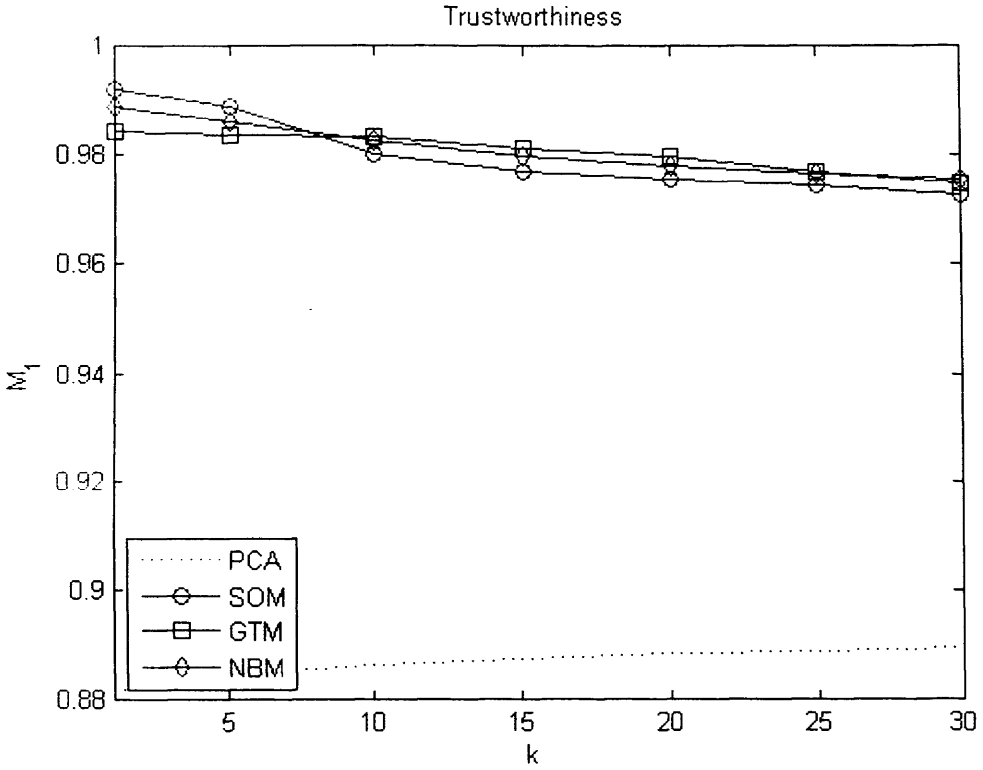

Figure 5.15(e) Trustworthiness measure for the Wood data set projection

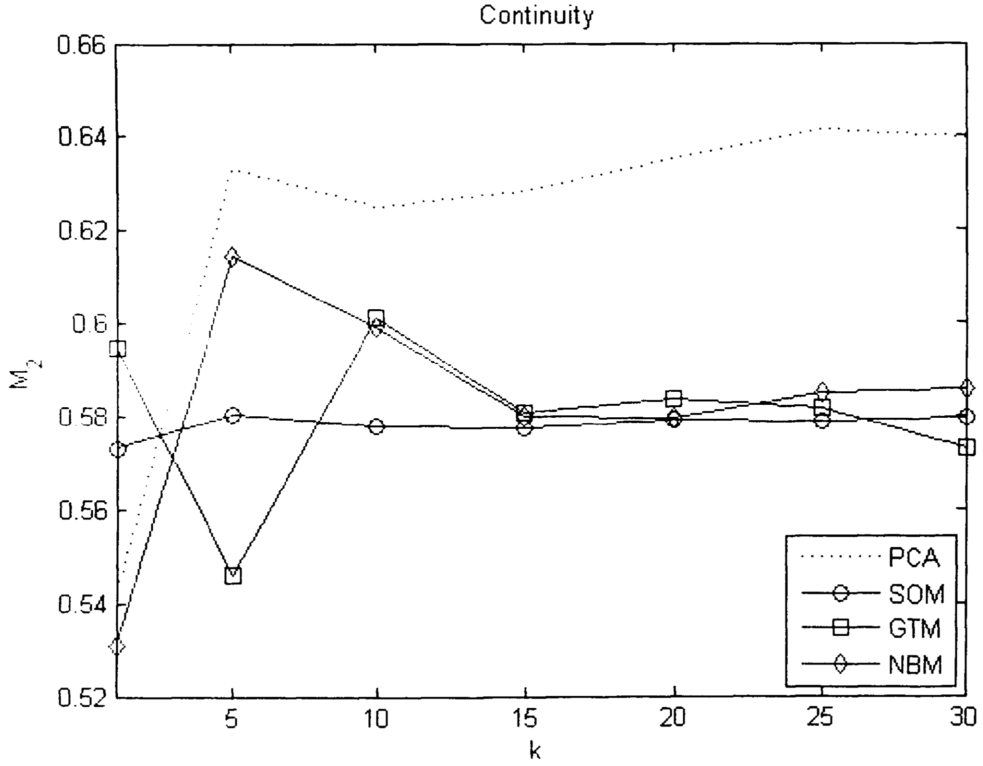

Figure 5.15(f) Continuity measure for the Wood data set projection

5.5 Relations with other algorithms

The NBM shares several characteristics with other mapping techniques. A brief comment expressing their relations is presented here.

In the SOM, each neuron in the output space has associated a weight vector in the input space, the NBM uses the same output neuron grid but each neuron has associated a naive Bayes model which is characterised by a mean vector, which acts as the SOM's weight vector, and also a variance vector. The NBM uses the posterior probability to identify the winning neuron while the SOM uses a distance metric (usually Euclidean). Like the SOM, the NBM updates the parameters for the winning neuron and its neighbours. This is achieved by replacing the posterior probability, which is computed using input space coordinates, in the updating rules with a neighbourhood function which considers the output space coordinates of the neurons.

The NBM is closely related to the GTM, because of its probabilistic nature. Both models define a probability density, uses the maximum likelihood to train the parameters, can perform posterior-mean projections, and have the log-likelihood function to monitor the convergence. The GTM projects the latent space variables into the input space and then performs the inverse process to obtain the data projection, while the NBM uses a discrete variable (hidden variable) in the input space to identify each naive Bayes model. Although the NBM must choose the neighbourhood function parameters for the topographic ordering, the GTM must choose an appropriate number of basis functions which has a direct influence in the final mapping results. In fact during the simulation section, it was found that for some data set the GTM was quite sensitive to the number of basis function, obtaining poor results in cases where the number of basis function selected for the training was not appropriate.

Van Hulle (2002) derived a learning algorithm for a mixture of kernels (Gaussian) by Kullback-Leibler divergence minimisation. The minimisation is achieved by invoking the Robbins-Monro stochastic approximation method (Robbins and Monro, 1951) which lead to similar updating rules as the ones used by NBM. The posterior probability in the updating rules is then changed by a neighbourhood function like the NBM, although Van Hulle uses an activity-based definition for the winner instead of the posterior probability-based definition used by the NBM.

Yin and Allison (2001) also derived a learning algorithm for a mixture of Gaussian and a mixture of Cauchy distributions by minimising the Kullback-Leibler divergence by means of the Robbins-Monro method. The density models are assumed to be heteroskedastic, like the NBM. The updating rules are limited to a small neighbourhood of the winning neuron, which has the largest posterior probability. So the neighbourhood function is in fact the posterior probability which is computed using input space coordinates, instead of the output space coordinates of the neurons used by the NBM, yielding different results in general.

5.6 Summary

A new learning algorithm for the naive Bayes model in order to obtain topographic maps has been presented. The learning algorithm is formulated under a probabilistic framework which enables the training of its parameters using maximum likelihood. Several examples were presented in order to assess the algorithms performance as well as comparison with well known mapping techniques. The results showed that the proposed method obtains good quality mapping visually and under the trustworthiness and continuity measures, capable of handling a good trade-off between these two measures. A key feature of the NBM is its natural inheritance of the naive Bayes characteristics (elegance, simplicity, and effectiveness) which enables a quick and simple implementation of this mapping technique.

Chapter 6

CONCLUSION

In this chapter the contributions and conclusions of this thesis are listed and suggestions for future work provided.

6.1 Contributions

The main contributions of this thesis are:

1. A new learning algorithm for the Bayesian network classifiers which incorporates the complexity of the network yielding structures that vary from no edges (naive Bayesian classifier) to $n-1$ edges (TAN model) where n is the number of features.

2. Proofs that the proposed learning algorithm minimises the cross-entropy between the real joint probability distribution and the approximation done by the Bayesian network, minimises the Bayes probability of error, and maximises the log-likelihood between the Bayesian network and the data.

3. A simple network encoding system which allows the definition of the *a priori* probability of a Bayesian network.

4. A new unsupervised learning algorithm for Bayesian network classifiers, which includes structure and parameter learning as well as the clustering of the input data.

5. The formulation of CL multinets, TAN, and the proposed Bayesian network classifier in chapter 3 into the Classification EM algorithm framework.

6. A new learning algorithm for the naive Bayes model in order to obtain topographic maps.

6.2 Conclusions

In this thesis the feasibility of Bayesian networks to carry out data mining tasks such as classification, clustering and high-dimensional data visualisation has been shown. New learning algorithms have been presented which improve the current state of the art of Bayesian networks in this field. The key conclusions for each topic analysed are:

- The complexity of the Bayesian network classifiers depends directly on the amount of data used for training. A spanning tree structure like the TAN model is therefore only justified when there is sufficient data. This is not a problem for the proposed method since it automatically regulates the number of edges depending on the amount of training data by means of a Bayesian monitoring system. The classification performance of the proposed Bayesian network classifier, compared to non-Bayesian classifiers using real-world problem data, was higher than that of the C4.5 and similar to that of a neural network.

- Testing the proposed unsupervised training method for three types of Bayesian network classifiers, using benchmark data sets as well as real-world data, showed that Bayesian network classifiers are promising models for clustering tasks, outperforming traditional clustering methods such as k-means and the standard EM algorithm. An important advantage of Bayesian network classifiers trained in an unsupervised way is that the resulting structure can give additional information on how the features are related (probabilistic dependencies) in each cluster, information which the traditional methods mentioned previously do not have. The proposed clustering technique can help a human expert identify and explain what each cluster means (represents) given the network representation for each cluster.

- It was found that naive Bayes model, the simplest Bayesian network, can be trained to generate topographic maps. Tests carried out on several data sets showed that the proposed method obtains good quality mapping visually and under topology preserving measures when compared to traditional linear and non-linear mapping techniques. An important advantage of the proposed mapping technique is its natural inheritance of the naive Bayes characteristics: elegance, simplicity, and effectiveness.

Although learning Bayesian networks from data is a difficult problem, this thesis has focused on developing simple models, by restricting the number of parents each node can have, while creating algorithms which are effective and practical. The competitive results obtained in all the experiments carried out as well as comparisons with popular data mining techniques should, hopefully, encourage more people to use Bayesian networks for data analysis and modelling applications.

6.3 Suggestions for future research

Possible extensions that can be made to the work presented in this thesis include:

- Developing algorithms that can handle continuous features. The algorithms described in chapters 3 and 4 consider only discrete variables. A known continuous probability distribution could be used to compute the marginal and conditional probabilities when computing the joint probability distribution encoded by the Bayesian network.

- Developing a method of determining the correct numbers of clusters for a given data set. The clustering algorithm using Bayesian networks presented in chapter 4 has the limitation that the user must specify beforehand the number of clusters. In order to obtain automatically the correct number of clusters during the training process, a Bayesian approach similar to the one presented in chapter 3 (used to determine the number of edges) could be adapted for the purpose of determining the correct number of clusters.

- Using more complex models to improve the quality of the output mapping. Chapter 5 presents the training of the naive Bayes model for topographic map formation. The use of more complex models like the TAN could be explored to see how this affects the quality of the output mapping. Nevertheless, the results should be significantly better than the naive Bayes model in order to accept a more complex learning algorithm than the one used for the NBM.

REFERENCES

Acid, S., de Campos, L. M., and Castellano, J. G. (2005). Learning Bayesian network classifiers: search in a space of partially directed acyclic graphs. *Machine Learning*, vol. 59, pp. 213-235.

Aitkenhead, M.J., and Aalders, I.H. (2007). Predicting land cover using GIS, Bayesian and evolutionary algorithm methods. *Journal of Environmental Management*, (Article in Press).

Alvarez, S.M., Poelstra, B.A., and Burd, R.S. (2006). Evaluation of a Bayesian decision network for diagnosing pyloric stenosis. *Journal of Pediatric Surgery*, vol. 41, no. 1, pp. 155-161.

Barash, Y., and Friedman, N. (2002). Context-specific Bayesian clustering for gene expression data. *Journal of Computational Biology*, vol. 9, pp. 169-191.

Bareither, C., and Luxhøj, J. T. (2007). Uncertainty and sensitivity analysis in bayesian belief networks: Applications to aviation safety risk assessment. *International Journal of Industrial and Systems Engineering*, vol. 2, no. 2, pp. 137-158.

Bezdek, J. C., and Pal, N. R. (1998). Some new indexes of cluster validity. *IEEE Trans. System Man Cybernetics Part B*, vol. 28, no. 3, pp. 301–315.

Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York.

Bishop, C. M., Svensén, M., and Williams, C. K. I. (1998). GTM: The generative topographic mapping. *Neural Comput.*, vol. 10, pp. 215–234.

Burger, L., and Van Nimwegen, E. (2008). Accurate prediction of protein-protein interactions from sequence alignments using a bayesian method. *Molecular Systems Biology*, 4:165.

Celeux, G., and Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition*, vol. 28, no. 5, pp. 781-793.

Celeux G., and Govaert, G. (1992). A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics & Data Analysis*, vol. 14, no. 3, pp. 315-332.

Cerquides, J., and Lopez de Mantaras, R. (2005). TAN classifiers based on decomposable distributions. *Machine Learning*, vol. 59, pp. 323-354.

Cheah, W.P., Kim, K.-Y., Yang, H.-J., Choi, S.-Y., and Lee, H.-J. (2007). A manufacturing-environmental model using Bayesian belief networks for assembly design decision support. *Lecture Notes in Computer Science*, 4570 LNAI, pp. 374-383.

Chen, Q., Li, G., Leong, T. Y., and Heng, C. K. (2007). Predicting coronary artery disease with medical profile and gene polymorphisms data. *Studies in Health Technology and Informatics*, vol. 129, no. 2, pp. 1219-1224.

Chen, R.-S., and Chang, C.-C. (2007). Using Bayesian networks to build data mining applications for a semiconductor cleaning process. *International Journal of Materials and Product Technology*, vol. 30, no. 4, pp. 386-407.

Cheng, J., and Greiner, R. (1999). Comparing Bayesian network classifiers. In Proceedings of the *15th Conf. on Uncertainty in Artificial Intelligence (UAI'99)*, CA: Morgan Kaufmann, San Francisco, pp. 101-107.

Chickering, D. M., and Heckerman, D. (1997). Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, vol. 29, n. 2-3, pp. 181-212.

Chickering, D.M. (1996). Learning Bayesian networks is NP-complete. In D. Fisher & A. Lenz, *Learning from data*, Springer-Verlag.

Chow, C. K., and Liu, C. N. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 462-467.

Cooper, G. F., and Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, vol. 9, pp. 309-347.

Cooper, G. F. (1990). The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, vol. 42, pp. 393-405.

Cover, T. M., and Thomas, J. A. (1991). *Elements of Information Theory*, New York: John Wiley & Sons.

Cowell, R. G., Verrall, R. J., and Yoon, Y. K. (2007). Modeling operational risk with bayesian networks. *Journal of Risk and Insurance*, vol. 74, no. 4, pp. 795-827.

Dash, D., and Cooper, G. F. (2002). Exact model averaging with naive Bayesian classifiers. Proc. *19th Int. Conf. Machine Learning (ICML '02)*, Sydney, Australia, pp. 91–98.

Dempster, A. N., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. R. Statist. Soc. Ser. B*, vol. 39, no. 1, pp. 1-38.

Deng, X., Geng, H., and Ali, H. H. (2006). Joint learning of gene functions - A bayesian network model approach. Journal of Bioinformatics and Computational Biology, vol. 4, no. 2, pp. 217-239.

Domingos, P., and Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, vol. 29, n. 2-3, pp. 103-130.

Dougherty, J., Kohavi, R., and Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. *Machine learning Proc. 12th Int. Conf.*, Morgan Kaufmann Publishers, San Francisco, CA, pp. 194-200.

Duda, R. O., and Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. NY: John Wiley & Sons, New York.

Estévez, P. A., Perez, C. A., and Goles, E. (2003). Genetic input selection to a neural classifier for defect classification of radiata pine boards. *Forest Prod. J.*, vol. 53, pp. 87-94.

Fayyad, U. M., and Irani, K. B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In Proceedings of the *13th Int. Joint Conf. on Artificial Intelligence*, CA: Morgan Kaufmann, San Francisco, pp. 1022-1027.

Friedman, J. (1997). On bias, variance, 0/1 - loss, and the curse-of dimensionality. *Data Mining and Knowledge Discovery*, vol. 1, pp. 55–77.

Friedman, N. (1998). The Bayesian structural EM algorithm. Cooper G. F., and Moral S., eds., Proc. *Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI '98)*, Morgan Kaufmann, San Francisco, CA, pp. 129-138.

Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, vol. 29, no. 2/3, pp. 131-163.

Granitto, P.M., Verdes, P.F., and Ceccatto, H.A. (2005). Large-scale investigation of weed seed identification by machine vision. *Computers and Electronics in Agriculture*, vol. 47, no. 1, pp. 15-24.

Grant, E. E., and Leavenworth, R. S. (1988). *Statistical quality control*, 6th edn., New York: McGraw Hill.

Greenberg, R., Cook, S. C., and Harris, D. (2005). A civil aviation safety assessment model using a bayesian belief network (BBN). *Aeronautical Journal*, vol. 109(1101), pp. 557-568.

Grossman, D., and Domingos, P. (2004). Learning Bayesian network classifiers by maximizing conditional likelihood. In Proceedings of the *21st Int. Conf. on Machine Learning*, ACM press, Banff, Canada, pp. 361-368.

Gurwicz, Y., and Lerner, B. (2006). Bayesian class-matched multinet classifier. SSPR/SPR, ser. *Lecture Notes in Computer Science*, 4109, pp. 145-153.

Heckerman, D. (1995). A tutorial on learning with Bayesian networks. *Technical Report MSR-TR-95-06*, Microsoft Research.

Heckerman, D., Geiger, D., and Chickering, D.M. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, vol. 20, no. 3, pp. 197-243.

Hellman, M. E., and Raviv, J. (1970). Probability of error, equivocation, and the Chernoff bound. *IEEE Trans. Inform. Theory*, vol. IT-16, pp. 368-372.

Hwang, K., and Zhang, B. (2005). Bayesian model averaging of Bayesian network classifiers over multiple node-orders: application to sparse datasets. *IEEE Trans. Syst., Man, and Cyber. B, Cyber.*, vol. 35, no. 6, pp. 1302-1310.

Im, S. -B., and Cho, S. -B. (2006). Context-based scene recognition using bayesian networks with scale-invariant feature transform. *Lecture Notes in Computer Science*, 4179 LNCS, pp. 1080-1087.

Infantes, G., Ingrand, F., and Ghallab, M. (2006). Learning behaviors models for robot execution control. In Proceedings of the *Sixteenth International Conference on Automated Planning and Scheduling (ICAPS 2006*), Cumbria, UK, pp. 394-397.

Jing, Y., Pavlovic, V., and Rehg, J. M. 2005. Efficient discriminative learning of Bayesian network classifier via Boosted Augmented Naive Bayes. In Proceedings of the *22nd Int. Conf. on Machine Learning*, Bonn, Germany, pp. 369-376.

John, G., and Kohavi, R. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, vol. 97, pp. 273-324.

Kaski, S., Nikkilä, J., Oja, M., Venna, J., Törönen, P., and Castrén E. (2003). Trustworthiness and metrics in visualizing similarity of gene expression," *BMC Bioinformatics*, 4:48.

Kass, R. E., and Raftery, A. E. (1995). Bayes factors. *Journal of American Statistical Association*, vol. 90, no. 430, pp. 773-795.

Keogh, E., and Pazzani, M. J. (2002). Learning the structure of augmented Bayesian classifiers. *International Journal on Artificial Intelligence Tools*, vol. 11, no. 4, pp. 587-601.

Kleiner, A., and Sharp, B. (2000). A new algorithm for learning Bayesian classifiers from data. In Proceedings of the *3rd IASTED International Conference on Artificial Intelligence and Soft Computing (ASC'2000)*, Banff, Canada, pp. 191-197.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proceedings of the *14th Int. Joint Conf. on Artificial Intelligence*, CA: Morgan Kaufmann, San Francisco, pp. 1137-1143.

Kohonen, T. (1995). *Self-Organizing Maps*, Springer- Verlag, Berlin, Germany.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biol. Cybern.*, vol. 43, pp. 59–69, 1982.

Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Am. Math. Soc.*, vol. 7, pp. 48-50.

Lam, W., and Bacchus, F. (1994). Learning Bayesian belief networks. An approach based on the MDL principle. *Computational Intelligence*, vol. 10, pp. 269-293.

Langley, P., and Sage, S. (1994). Induction of selective Bayesian classifiers. In Proceedings of the *Tenth Conference on Uncertainty in Artificial Intelligence*, R. Lopez de Mantaras and D. Poole Eds., CA: Morgan Kaufmann, San Francisco, , pp. 399–406.

Langley, P., Iba, W., and Thompson, K. (1992). An analysis of Bayesian classifiers. In Proceedings of the *Tenth National Conf. on Artificial Intelligence (AAAI'92)*, Menlo Park, CA, pp. 223-228.

Lazkano, E., Sierra, B., Astigarraga, A., and Martínez-Otzeta, J. M. (2007). On the use of bayesian networks to develop behaviours for mobile robots. *Robotics and Autonomous Systems*, vol. 55, no. 3, pp. 253-265.

Li, J., and Shi, J. (2007). Knowledge discovery from observational data for process control using causal Bayesian networks. *IIE Transactions*, vol. 39, no. 6, pp. 681-690.

Lu, J., Bai, C., and Zhang, G. (2007). E-service cost benefit evaluation and analysis. *Studies in Computational Intelligence*, vol. 37, pp. 389-409.

Lucas, P. J. F., van der Gaag, L. C., and Abu-Hanna, A. (2004). Bayesian networks in biomedicine and health-care. *Artif Intell Med*, vol. 30, no. 3, pp. 201-214.

Luo, F., Chen, X., and Gu, B. (2005). Formation mechanism analysis of aviation calamity based on bayesian network. *Beijing Hangkong Hangtian Daxue Xuebao/Journal of Beijing University of Aeronautics and Astronautics*, vol. 31, no. 8, pp. 934-938.

Ma, J., and Dai, Q. (2005). Migratory Locust Hazard monitoring and prediction using the Bayesian network inference. In Proceedings of the *International Geoscience and Remote Sensing Symposium (IGARSS'05)*, vol. 5, pp. 3623-3626.

MacKay, D. J. C. (2003). *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, UK.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In Proceedings of the *Fifth Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, Berkeley, 1, pp. 281-297.

Mani, S., Valtorta, M., and McDermott, S. (2005). Building Bayesian network models in medicine: The MENTOR experience. *Applied Intelligence*, vol. 22, no. 2, pp. 93-108.

Meilă, M., and Jaakkola, T. (2000). Tractable Bayesian learning of tree belief networks. In Proceedings of the *16th Conf. on Uncertainty in Artificial Intelligence*, CA: Morgan Kaufmann, San Francisco, pp. 380-388.

Meilă, M., and Jordan, M. I. (2000). Learning with mixtures of trees. *Journal of Machine Learning Research*, vol. 1, pp. 1-48.

Meilă, M., and Heckerman, D. (1998). An Experimental Comparison of Several Clustering and Initialization Methods. Microsoft Research Tech. Rep. MSR-TR-98-06.

McMahon, S. M. (2005). Quantifying the community: Using bayesian learning networks to find structure and conduct inference in invasions biology. *Biological Invasions*, vol. 7, no. 5, pp. 833-844.

Monti, S., and Cooper, G. (1999). A Bayesian network classifier that combines a finite mixture model and a naive-Bayes model. In Proceedings of the *15th Conf. on Uncertainty in Artificial Intelligence (UAI'99)*, CA: Morgan Kaufmann, San Francisco. pp. 447-456.

Mukhopadhyay, A., Chatterjee, S., Saha, D., Mahanti, A., and Sadhukhan, S. K. (2006). E-risk management with insurance: A framework using copula aided bayesian belief networks. HICSS '06. Proceedings of the *39th Annual Hawaii International Conference on System Sciences*, Hawaii, USA, pp. 126a.

Murphy, P. M., and Aha, D. W. UCI repository of machine learning databases. http://www.ics.uci.edu/~mlearn/MLRepository.html

Needham, C. J., Bradford, J. R., Bulpitt, A. J., Care, M. A., and Westhead, D. R. (2006). Predicting the effect of missense mutations on protein function: Analysis with bayesian networks. *BMC Bioinformatics*, 7:405.

Neil, M., Fenton, N., and Tailor, M. (2005). Using bayesian networks to model expected and unexpected operational losses. *Risk Analysis*, vol. 25, no. 4, pp. 963-972.

Neapolitan, R. E. (2004). *Learning Bayesian Networks*. NJ: Pearson Prentice Hall, Upper Saddle River.

Nikolajewa, S., Pudimat, R., Hiller, M., Platzer, M., and Backofen, R. (2007). BioBayesNet: A web server for feature extraction and bayesian network modeling of biological sequence data. *Nucleic Acids Research*, vol. 35, no. suppl_2, pp. W688-W693.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, CA: Morgan Kaufmann, San Francisco.

Peña, J. M., Lozano J. M., and Larrañaga P. (2004). Unsupervised learning of Bayesian networks via estimation of distribution algorithms: an application to gene expression data clustering. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 12(Supplement-1), pp. 63-82.

Peña, J. M., Lozano J. M., and Larrañaga P. (2000). An improved Bayesian structural EM algorithm for learning Bayesian networks for clustering. *Pattern Recognition Letters*, vol. 21, no. 8, pp. 779-786.

Peña, J. M., Lozano J. M., and Larrañaga P. (1999). Learning Bayesian networks for clustering by means of constructive induction. *Pattern Recognition Letters*, vol. 20, no. 11-13, pp.1219-1230.

Perzyk, M., Biernacki, R., and Kochański, A. (2005). Modeling of manufacturing processes by learning systems: The naive Bayesian classifier versus artificial neural networks. *Journal of Materials Processing Technology*, vol. 164-165, pp. 1430-1435.

Pham, D. T., and Alcock R. J. (2003). *Smart Inspection Systems*, Academic Press, London.

Pham, D. T., and Oztemel, E. (1992). Control chart pattern recognition using neural networks. *J. Sys. Eng.*, vol. 2, pp. 256-262.

Pudimat, R., Schukat-Talamazzini, E. -G., and Backofen, R. (2005). A multiple-feature framework for modelling and predicting transcription factor binding sites. *Bioinformatics*, vol. 21, no. 14, pp. 3082-3088.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*, CA: Morgan Kaufmann, San Francisco.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, vol. 14, pp. 465-471.

Roberts, S. J., Everson, R., and Rezek I. (2000). Maximum Certainty Data Partitioning. *Pattern Recognition*, vol. 33, no. 5, pp. 833-839.

Robbins, H., and Monro, S. (1951). A stochastic approximation method. *Ann. Math. Statist.*, vol. 22, pp. 400-407.

Rodin, A. S., and Boerwinkle, E. (2005). Mining genetic epidemiology data with bayesian networks I: Bayesian networks and example application (plasma apoE levels). *Bioinformatics*, vol. 21, no. 15, pp. 3273-3278.

Ruz, G. A., Estévez, P. A., and Ramírez, P. A. (2007). Automated visual inspection system for wood defect classification using computational intelligence techniques. *International Journal of Systems Science* (accepted for publication).

Ruz, G. A., Estévez, P. A., and Perez, C. A. (2005). A neurofuzzy color image segmentation method for wood surface defect detection. *Forest Prod. J.*, vol. 55, pp. 52-58.

Santafé G., Lozano, J. A., and Larrañaga, P. (2006a). Bayesian model averaging of naive Bayes for clustering. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, vol. 36, no. 5, pp. 1149-1161.

Santafé G., Lozano, J. A., and Larrañaga, P. (2006b). Bayesian model averaging of TAN models for clustering. *European Workshop on Probabilistic Graphical Models (PGM 2006)*, Prague, Czech Republic, pp. 271-278.

Shi, H., and Huang, H. (2002). Learning tree-augmented naïve Bayesian networks by reduced space requirements. In Proceedings of the *First Int. Conf. on Machine Learning and Cybernetics*, Beijing, pp. 1232-1236.

Suebnukarn, S., and Haddawy, P. (2006). A Bayesian approach to generating tutorial hints in a collaborative medical problem-based learning system. *Artificial Intelligence in Medicine*, vol. 38, no. 1, pp. 5-24.

Sun, L., and Shenoy, P. P. (2007). Using bayesian networks for bankruptcy prediction: Some methodological issues. *European Journal of Operational Research*, vol. 180, no. 2, pp. 738-753.

Titterington, D.M. (1984). Recursive parameter estimation using incomplete data. *J. Royal Statistical Soc., Series B (Methodological)*, vol. 2, no. 46, pp. 257-267.

Ueda, N., and Nakano, R. (1998). Deterministic annealing EM algorithm. *Neural Networks*, vol. 11, pp.271–282.

Van Hulle, M. M. (2002). Joint entropy maximization in kernel-based topographic maps. *Neural Computat.*, vol. 14, no. 8, pp. 1887–1906.

Venna, J., and Kaski, S. (2006). Local multidimensional scaling. *Neural Networks*, vol. 19, pp. 889-899.

Venna, J., and Kaski, S. (2005). Local multidimensional scaling with controlled tradeoff between trustworthiness and continuity. In Proceedings of *WSOM'05, 5th Workshop On Self-Organizing Maps*, Paris, France, pp. 695–702.

Venna, J., and Kaski, S. (2001). Neighborhood preservation in nonlinear projection methods: An experimental study. in Proceedings of *ICANN 2001*, G. Dorffner, H. Bischof, and K. Hornik, Eds., Berlin, Germany, pp. 485-491.

Verduijn, M., Peek, N., Rosseel, P.M.J., de Jonge, E., and de Mol, B.A.J.M. (2007). Prognostic Bayesian networks. I: Rationale, learning procedure, and clinical use. *Journal of Biomedical Informatics*, vol. 40, no. 6, pp. 609-618.

Visscher, S., Lucas, P., Bonten, M., and Schurink, K. Improving the therapeutic performance of a medical Bayesian network using noisy threshold models. (2005). *Lecture Notes in Computer Science*, 3745 LNBI, pp. 161-172.

Wang, H., and Wang, J. (2006). A quantitative diagnostic method based on Bayesian networks in traditional Chinese medicine. *Lecture Notes in Computer Science*, 4234 LNCS - III, pp. 176-183.

Wang, H., and Zong, X. (2006). A new computerized method for tongue classification. In Proceedings - ISDA 2006: *Sixth International Conference on Intelligent Systems Design and Applications*, Jinan, Shandong, China, vol. 2, pp. 508-511.

Wang, X. R., and Ramos, F. T. (2005). Applying structural em in autonomous planetary exploration missions using hyperspectral image spectroscopy. In Proceedings of the *2005 IEEE International Conference on Robotics and Automation (ICRA 2005),* Barcelona, Spain, pp. 4284-4289.

Weber, P., and Jouffe, L. (2006). Complex system reliability modelling with Dynamic Object Oriented Bayesian Networks (DOOBN). *Reliability Engineering and System Safety*, vol. 91, no. 2, pp. 149-162.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Yu, P. S., Zhou, Z., Steinbach, M., Hand, D. J., and Steinberg D. (2008). Top 10 algorithms in data mining. *Knowl Inf Syst*, vol. 14, pp. 1-37.

Yin, H., and Allinson, N. M. (2001). Self-organizing mixture networks for probability density estimation. *IEEE Trans. Neural Networks*, vol. 12, pp. 405–411.

Yuan, Y., Guo, L., Shen, L., and Liu, J. S. (2007). Predicting gene expression from sequence: A reexamination. *PLoS computational biology*, e243, vol. 3, no. 11, pp. 2391-2397.

Zhu, W.-F., Yan, J.-F., and Huang, B.-Q. (2006). Application of Bayesian network in syndrome differentiation system of traditional Chinese medicine. *Journal of Chinese Integrative Medicine*, vol. 4, no. 6, pp. 567-571.

Zou, F., Li, C., Hu, X., and Zhou, C. (2007). Combination of principal component analysis and Bayesian network and its application on syndrome classification for chronic gastritis in traditional Chinese medicine. Proceedings of the *Third International Conference on Natural Computation, ICNC 2007*, Haikou, China, pp. 588-592.