Departamento de Ciencias de la Computación e Inteligencia Artificial
Konputazio Zientziak eta Adimen Artifiziala Saila
Department of Computer Science and Artificial Intelligence

Universidad del País Vasco
Euskal Herriko Unibertsitatea
University of the Basque Country

Learning Bayesian Networks from Data with Factorisation and Classification Purposes. Applications in Biomedicine

by

Rosa Blanco

Dissertation submitted to the Department of Computer Science and Artificial Intelligence of the University of the Basque Country in partial fulfilment of the requirements for the PhD degree in Computer Science

Donostia - San Sebastián, March 2005

Contents

List of Figures

List of Tables

Acknowledgments

There are some people to whom I am deeply indebted for their help and support in the past years throughout the long process that ends with this dissertation. I would like to make the use of this opportunity to acknowledge their never faltering encouragement.

First, I wish to thank Pedro Larrañaga and Iñaki Inza, my thesis advisors, for everything they have taught me in the last few years. Special thanks for their wise supervision, support, encouragement and example in the research field.

I also owe a debt of gratitude to my colleagues of the Intelligent Systems Group at the Department of Computer Science and Artificial Intelligence. I will always affectionately remember my labmates during the last four years.

I am grateful to the University of the Basque Country for their financial support during these years under grant 9/UPV/EHU 00140.226-12084/2000.

I would like to thank Linda van der Gaag and the rest of the people of the Decision Support Systems group at Utrecht University (The Netherlands) for their interest in my work and for hosting me in spring 2004. The results of their collaboration are included in this dissertation.

My gratitude to Marisa Merino, physician of the Servicio Vasco de Salud-Osakidetza. Her suggestions for applying Bayesian classifiers to real biomedical problems have provided me with another point of view.

Finally, my deepest affection to my family for giving me an education and even understanding me. All my love to Xabi, who has always supported me from the very start to the very end of this adventure.

Part I

Introduction

1

Introduction

For the past years, probabilistic graphical models Howard and Matheson, 1981; Pearl, 1988; Lauritzen, 1996 have become a powerful representation to encode uncertain knowledge in expert systems. Bayesian networks Castillo et al., 1997; Jensen, 2001; Neapolitan, 2003 are popular probabilistic graphical models whose use has grown spectacularly.

A Bayesian network has two components: a network structure and a set of conditional probabilities. The network structure is a directed acyclic graph which depicts the relationships between the variables or nodes using arrows. The conditional probability measures represent the uncertainty of the domain.

When there is no expert to trace the probabilistic relationship of a domain, the automatic learning of a Bayesian network from a set of samples is a useful and widely accepted alternative. The resulting structure reflects the conditional (in)dependencies between the domain variables. The first automatic approaches produce a list of conditional (in)dependencies by means of statistical test. However, another automatic approach has appeared: the score+search method. In a score+search framework, an intelligent search in a specific search space is performed, assessing each Bayesian network structure proposed by a scoring measure. The main focus of this approach is to find the network structure which best fits the dataset.

On the other hand, the supervised classification task, a field of the machine learning area, tries to learn a classifier to label a set of unseen instances. In order to perform this issue, a set of samples with a special variable, the class node, is used. Bayesian networks could be learnt to solve a supervised classification problem. Several Bayesian classifiers have been proposed in the literature depending on the limitations imposed on a Bayesian network.

Nevertheless, the inclusion of non-appropriate variables could worsen the performance of the Bayesian classifiers. In fact, the Bayesian classification models are strongly influenced by the addition of redundant variables. Thus, the ideas of the feature subset selection task are adapted to reject these variables in the final Bayesian classification model. The filter and wrapper methods have been proposed for feature reduction. The filter approach indirectly

distinguishes between the class variables by looking at the intrinsic characteristics of the dataset by means of a scoring function. However, due to the increase in computational sources and the work of John et al. (1994), *wrapper* methods are developed. This approach focusses on the characteristics of the classifiers to discriminate between the class values. Then, in the wrapper approach the classification models are used like a black box.

Real medical classification problems are a challenge to the machine learning field. When learning a reliable classifier, the relationship between the number of variables and the number of samples is usually not well balanced. Moreover, due to the collection process, real medical datasets often suffer from a high ratio of missing values. Nevertheless, once a Bayesian classifier is learnt, it provides the medical staff with more simplicity and understanding about the medical problem.

1.1 Overview of the dissertation

This dissertation is divided into four parts and is made up fourteen chapters. Part I consists of three chapters. This chapter is an introduction to the dissertation. Chapter 2 introduces the probabilistic graphical models with special attention on Bayesian networks. The basic notation used is presented and some relevant topics reviewed. Chapter 3 formalises the terminology of the optimisation task. In order to place the optimisation task in this work, classical optimisation methods are visited.

Part II focusses on the learning of Bayesian network structures. In Chapter 4, the learning of Bayesian network from data is formulated as a score+search approach. The methods for Bayesian network induction are reviewed. These methods are algorithms based on conditional independence tests, score+search methods and hybrid algorithms, which have emerged during the last years. Chapter 5 presents the novel score+search algorithms proposed to perform the search of Bayesian network structures. The floating algorithms and the GRASP method are introduced and adapted to Bayesian network induction. Empirical experimentation performed over well-known databases is included. Finally, this part concludes with Chapter 6, which presents a set of conclusions of the work carried out work in the area of Bayesian network structure learning, together with future work lines.

Part III is dedicated to the supervised classification task. In detail, it deals with the novel filter and wrapper approaches to Bayesian classifier induction. Chapter 7 introduces the concepts required to perform the supervised classification task by means of Bayesian classifiers. The supervised classification task, feature subset selection and accuracy estimation are presented. In Chapter 8, the Bayesian classification models (naive Bayes, seminaive Bayes, tree augmented naive Bayes and k dependence Bayesian classifiers), which are the main focus of the filter and wrapper approaches, are exposed. In Chapter 9 the novel filter and wrapper approaches to Bayesian classifier induction are

presented. First, the state of the art of both approaches is set and, then, the novel filter and wrapper approaches are proposed. In order to check the advantages of the novel methods, an empirical experimentation is carried out over synthetic datasets and well-known UCI Blake and Merz, 1998 datasets. This Part concludes with Chapter 10, where a set of conclusions about the filter and wrapper methods to learn Bayesian classification models and future work lines are revealed.

Finally, Part IV introduces three possible applications of Bayesian classifiers to medical domains. Chapter 11 is dedicated to predict the survival of cirrhotic patients within six months after a non-surgical method resulting in decompression of the portal system. The patient database is collected from the Clínica Universitaria de Navarra and is composed of medical findings. The reliability of the classifiers and the satisfaction of the medical staff are crucial in this biomedical application. In Chapter 12, the identification of the oesophageal carcinoma type is the task of the Bayesian classifiers. A database provided by the Netherlands Cancer Institute is used to perform this task. However, this database is sparse and some problems to apply the novel Bayesian classification models proposed in Chapter 9 to it appears. Chapter 13 is related to a growing area of genomics: DNA microarrays. The Bayesian classifiers are applied to two well-known DNA microarray databases with promising results. These results are supported by the literature in the field. At the end, as in the other parts, Chapter 14 presents a set of conclusions about the medical applications of the Bayesian classifiers and future work lines.

2

Probabilistic graphical models

The probabilistic graphical models Howard and Matheson, 1981; Pearl, 1988; Lauritzen, 1996 are a powerful representation to encode uncertain knowledge in expert systems. It has been very popular for the last years. The paradigm is introduced in this chapter in order to explain the general aspects used to perform this work. Once a general notation and the probabilistic graphical model paradigm are presented, the focus is on a well-known probabilistic graphical model: the Bayesian networks, a crucial tool for the development of this work. This chapter is an adaptation of Larrañaga (2001).

2.1 Terminology and basic concepts

In order to standardise the notation used in this work, some basic concepts should be set.

Let X_i be a random variable. An instance of X_i is denoted x_i. $p(X_i = x_i)$ or $p(X_i)$ represents the *generalised probability distribution* DeGroot, 1970 over point x_i. In the same way, $\boldsymbol{X} = (X_1, \ldots, X_n)$ denotes a n-dimensional random variable, and $\boldsymbol{x} = (x_1, \ldots, x_n)$ denotes one of its feasible instances. The *joint generalised probability distribution* of $\boldsymbol{x}$ is represented as $\rho(X_i = x_i)$ or $\rho(x_i)$. The *generalised conditional probability distribution* of variable X_i given the value x_j of variable X_j is represented as $\rho(X_i = x_i|X_j = x_j)$ or $\rho(x_i|x_j)$. The symbol D denotes the dataset, i.e. a set of N instances of variable $\boldsymbol{X} = (X_1, \ldots, X_n)$.

If random variable X_i is discrete, $\rho(X_i = x_i) = p(X_i = x_i)$ or $p(x_i)$ is known as the *mass probability* for variable X_i. If all variables in $\boldsymbol{X}$ are discrete, $\rho(\boldsymbol{X} = \boldsymbol{x}) = p(\boldsymbol{X} = \boldsymbol{x}))$ or $p(\boldsymbol{x})$ is called the *joint probability mass* and $\rho(X_i = x_i|X_j = x_j) = p(X_i = x_i|X_j = x_j)$ or $p(x_i|x_j)$ is the *conditional mass probability* of variable X_i given that $X_j = x_j$.

If random variable X_i is continuous, $\rho(X_i = x_i) = f(X_i = x_i)$ or $f(x_i)$ is the *density function* of X_i. If all variables in $\boldsymbol{X}$ are continuous, $\rho(\boldsymbol{X} = \boldsymbol{x}) = f(\boldsymbol{X} = \boldsymbol{x})$ or $f(\boldsymbol{x})$ is known as the *joint density function* and $\rho(X_i =$

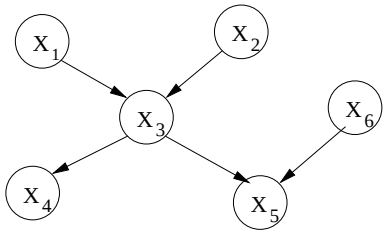


Fig. 2.1. Structure for a probabilistic graphical model defined over $\boldsymbol{X} = (X_1, X_2, X_3, X_4, X_5, X_6)$.

$x_i|X_j = x_j) = f(X_i = x_i|X_j = x_j)$ or $f(x_i|x_j)$ denotes the *conditional density probability* of the variable X_i given that $X_j = x_j$.

Let $\boldsymbol{X} = (X_1, \ldots, X_n)$ be a vector variable. Then, x_i denotes a value of X_i, the i-th component of $\boldsymbol{X}$ and $\boldsymbol{y} = (x_i)_{X_i \in \boldsymbol{Y}}$denotes a value of $\boldsymbol{Y} \subseteq \boldsymbol{X}$. A *probabilistic graphical model* for $\boldsymbol{X}$ is a graphical factorisation of the joint generalised probability distribution, $\rho(\boldsymbol{X} = \boldsymbol{x})$ or $\rho(\boldsymbol{x})$. The representation of a probabilistic graphical model has two components: a structure and a set of local generalised probability distributions. The structure S for $\boldsymbol{X}$ is a *directed acyclic graph* which represents a set of conditional (in)dependence relationships Dawid, 1979 between the variables of $\boldsymbol{X}$.

The structure S for $\boldsymbol{X}$ represents for each variable the asserts that X_i and $\{X_1, \ldots, X_n\} \setminus \boldsymbol{Pa}_i^S$ are independent given $\boldsymbol{Pa}_i^S$, $i = 2, \ldots, n$, where $\boldsymbol{Pa}_i^S$ is the set of parents of variable X_i in the structure S. Therefore, the factorisation applying the chain rule results is as follows:

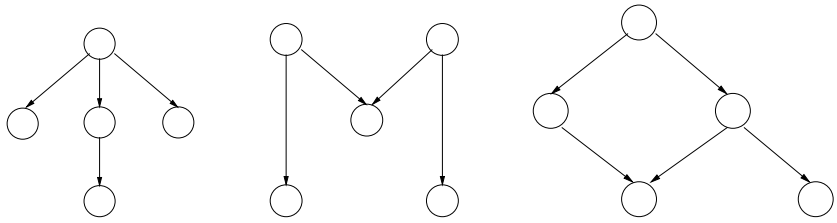
$$\begin{aligned}
\rho(\boldsymbol{x}) &= \rho(x_1, \ldots, x_n) \\
&= \rho(x_1)\rho(x_2|x_1) \ldots \rho(x_i|x_1, \ldots, x_{i-1}) \ldots \rho(x_n|x_1, \ldots, x_{n-1}) \\
&= \rho(x_1)\rho(x_2|\boldsymbol{pa}_2^S) \ldots \rho(x_i|\boldsymbol{pa}_i^S) \ldots \rho(x_n|\boldsymbol{pa}_n^S) \\
&= \prod_{i=1}^{n} \rho(x_i|\boldsymbol{pa}_i^S) \qquad (2.1)
\end{aligned}$$

In this representation, it is assumed that, for each variable, the local generalised probability distribution depends on a finite set of parameters $\boldsymbol{\theta}_S \in \boldsymbol{\Theta}_S$. Hence, equation 2.1 could be rewritten as:

$$\rho(\boldsymbol{x}|\boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i|\boldsymbol{pa}_i^S, \boldsymbol{\theta}_i)$$

where $\boldsymbol{\theta}_S = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n)$. Taking both components of the probabilistic graphical model into account, this will be represented by $M = (S, \boldsymbol{\theta}_S)$, i.e. a pair formed by the structure and its associated parameters.

Example 2.1 *The structure of the probabilistic graphical model represented in Figure 2.1 provides the following factorisation of the joint generalised probability distribution:*



(a) Tree (b) Polytree (c) Multiply connected

Fig. 2.2. Different degrees of complexity in the structure of probabilistic graphical models.

$$\rho(x_1, x_2, x_3, x_4, x_5, x_6) = \rho(x_1)\rho(x_2)\rho(x_3|x_1, x_2)\rho(x_4|x_3)\rho(x_5|x_3, x_6)\rho(x_6)$$

Informally, an arc between two nodes relates the two nodes so that the value of the variable corresponding to the ending node of the arc depends on the value of the variable corresponding to the starting node.

In order to understand the underlying semantics of probabilistic graphical models, the conditional independence and separation criterion become crucial. To check the conditional independence between variables $\boldsymbol{Y}$, $\boldsymbol{Z}$ given $\boldsymbol{W}$, the smallest subgraph containing $\boldsymbol{Y}$, $\boldsymbol{Z}$ and $\boldsymbol{W}$ should be considered. The moralisation of this subgraph, which is carried out by adding an edge between parents with common children and then turning the arcs into edges, is required. If in the undirected graph obtained every path between the variables in $\boldsymbol{Y}$ and variables in $\boldsymbol{Z}$ is blocked by a variable in $\boldsymbol{W}$, the variables $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are conditionally independent given $\boldsymbol{W}$ in the original graph. Otherwise, the variables $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are considered conditionally dependent given $\boldsymbol{W}$.

Depending on the connectivity of the model structure, several degrees of complexity –see Figure 2.2– in probabilistic graphical models can be considered:

- Without dependencies
  No variable has a parent variable. Therefore, the following factorisation is attained:
  $$\rho(\boldsymbol{x}|\boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i|\boldsymbol{\theta}_i).$$
- Tree
  Each variable has one parent at most. Thus, the factorisation is as follows:
  $$\rho(\boldsymbol{x}|\boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i|x_{j(i)}, \boldsymbol{\theta}_i)$$
  where $X_{j(i)}$ is the (possibly empty) parent of X_i.
- Polytree

$$\rho(\boldsymbol{x}|\boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i|x_{j_1(i)}, \ldots, x_{j_r(i)}, \boldsymbol{\theta}_i)$$

where $\{X_{j_1(i)}, \ldots, X_{j_r(i)}\}$ is the (possibly empty) set of parents of X_i in the structure and the parents of each variables are mutually independent:

$$\rho(x_{j_1(i)}, \ldots, x_{j_r(i)}) = \prod_{k=1}^{n} \rho(x_{j_k(i)}), \forall i = 1, \ldots, n.$$

- Multiply connected
  While in tree and polytree structures there is one path connecting two given nodes in the directed acyclic graph, in multiply connected structures two nodes could be connected by more than one path. Consequently, the factorisation is as follows:

$$\rho(\boldsymbol{x}|\boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i|\boldsymbol{pa}_i^S, \boldsymbol{\theta}_i).$$

2.2 Bayesian networks

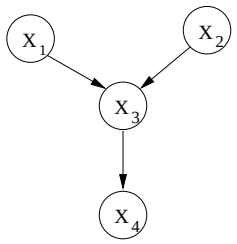
Bayesian networks are directed acyclic graphs where the nodes are random variables with a discrete set of values and the arcs specify the (in)dependence assumptions which should be held between the random variables. For the past years, the interest in Bayesian networks has grown spectacularly. As a result of this interest, a large number of theoretical and practical publications has appeared. The book of Pearl (1988) is a classic and well-known reference. The basic ideas of propagation algorithms are explained in Neapolitan (1990) and a study in detail is presented in Shafer (1996). A recommendable tutorial introduction could be found in Castillo et al. (1997), Jensen (2001) and Neapolitan (2003). The mathematical analysis of graphical models is provided in Lauritzen (1996).

Bayesian networks are a popular probabilistic framework for reasoning under uncertainty. The propagation of the evidence through the model depends on the structure reflecting the conditional (in)dependencies between the variables.

2.2.1 Notation for Bayesian networks

In a Bayesian network structure, when the discrete variable $X_i \in \boldsymbol{X}$ has r_i possible values, $x_i^1, \ldots, x_i^{r_i}$, the local distribution, $p(x_i|\boldsymbol{pa}_i^{j,S}, \boldsymbol{\theta}_i)$ is an unrestricted discrete distribution:

$$p(x_i^k|\boldsymbol{pa}_i^{j,S}, \boldsymbol{\theta}_i) = \theta_{x_i^k|\boldsymbol{pa}_i^j} \equiv \theta_{ijk}$$



Local probabilities

$\boldsymbol{\theta}_1 = (\theta_{1-1}, \theta_{1-2})$ $p(x_1^1), p(x_1^2)$

$\boldsymbol{\theta}_2 = (\theta_{2-1}, \theta_{2-2}, \theta_{2-3})$ $p(x_2^1), p(x_2^2), p(x_2^3)$

$\boldsymbol{\theta}_3 = (\theta_{311}, \theta_{321}, \theta_{331},$ $p(x_3^1|x_1^1, x_2^1), p(x_3^1|x_1^1, x_2^2), p(x_3^1|x_1^1, x_2^3),$

$\theta_{341}, \theta_{351}, \theta_{361},$ $p(x_3^1|x_1^2, x_2^1), p(x_3^1|x_1^2, x_2^2), p(x_3^1|x_1^2, x_2^3),$

$\theta_{312}, \theta_{322}, \theta_{332},$ $p(x_3^2|x_1^1, x_2^1), p(x_3^2|x_1^1, x_2^2), p(x_3^2|x_1^1, x_2^3),$

$\theta_{342}, \theta_{352}, \theta_{362})$ $p(x_3^2|x_1^2, x_2^1), p(x_3^2|x_1^2, x_2^2), p(x_3^1|x_1^2, x_2^3),$

$\boldsymbol{\theta}_4 = (\theta_{411}, \theta_{421}, \theta_{412}, \theta_{422})$ $p(x_4^1|x_3^1), p(x_4^1|x_3^2), p(x_4^2|x_3^1), p(x_4^2, x_3^2$

Factorisation of the joint mass probability

$p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2)p(x_3|x_1, x_2)p(x_4|x_3)$

Fig. 2.3. Structure, local probabilities and resulting factorisation for a Bayesian network with 4 variables (X_1, X_3, and X_4 with two possible values and X_2 with 3 possible values).

variable X_i	*possible values* r_i	*set of parents* $\boldsymbol{Pa}_i$	*possible values of parents* q_i
X_1	2	$\emptyset$	0
X_2	3	$\emptyset$	0
X_3	2	$\{X_1, X_2\}$	6
X_4	2	$\{X_3\}$	2

Table 2.1. Variables (X_i), number of possible values of variable (r_i), set of parents of a variable ($\boldsymbol{Pa}_i$) and number of possible configurations of the parents (q_i).

where $\boldsymbol{pa}_i^{1,S}, \ldots, \boldsymbol{pa}_i^{q_i,S}$ denotes the values of $\boldsymbol{Pa}_i^S$, the set of parents of X_i in the structure S. q_i is the number of possible different instances of the parent variables of X_i, hence $q_i = \prod_{X_g \in \boldsymbol{Pa}_i^S} r_g$.

The local parameters are given by $\boldsymbol{\theta}_i = \left((\theta_{ijk})_{k=1}^{r_i}\right)_{j=i}^{q_i}$. In other words, parameter θ_{ijk} is the conditional probability of variable X_i being in its k-th state given that the set of parents are in its j-th configuration.

Example 2.2 *In order to understand the notation, Figure 2.3 and Table 2.1 are presented. The values shown in Table 2.1 are obtained from Figure 2.3. The number of parameters required to specify the joint probability mass is reduced from* 23 *to* 11 *due to the factorisation of the joint probability mass derived from the structure of the Bayesian network in Figure 2.3.*

In order to assess a Bayesian network, it is required to set:

- A structure by means of a directed acyclic graph which reflects the set of conditional (in)dependencies between the variables.
- The unconditional probabilities for all nodes without parents, $p(X_i^k|\emptyset, \boldsymbol{\theta}_i)$ or θ_{i-k}.
- The conditional probabilities for all nodes with parents given all the possible configurations of their direct predecessors, $\theta_{ijk} = p(x_i^k|\boldsymbol{pa}_i^{j,S}, \boldsymbol{\theta}_i)$.

PLS
Find an ancestral ordering π of the nodes of the Bayesian network
repeat for $j = 1, 2, \ldots, N$
repeat for $i = 1, 2, \ldots, n$
$x_{\pi(i)} \leftarrow$ generate a value from $p(x_{\pi(i)}|\boldsymbol{pa}_{\pi(i)})$

Fig. 2.4. Pseudocode for the PLS method.

2.2.2 Model induction

Once the Bayesian network is built, it is an efficient tool to perform probabilistic inference. However, the problem of learning the network remains. The structure and conditional probabilities necessary to characterise the Bayesian network could be provided either by experts or by automatic induction from a database. On the other hand, the learning procedure could be divided into *structural learning* and *parametric learning*.

The development of the databases in general during the past years facilitates the access to a large number of cases. Due to this accessibility, the learning algorithms proposed have increased. This subject is one of the topics of this dissertation, hence the model induction of Bayesian networks is deeply analysed in Chapter 4.

2.2.3 Simulation

The simulation of Bayesian networks could be considered an alternative to the exact propagation methods developed to reason with the networks.

A large number of approximations to Bayesian network simulation has been developed. Some of these approaches are the *likelihood weighting method* developed independently by Fung and Chang (1990) and Shachter and Peot (1990) and analysed by Shwe and Cooper (1991), the *backward-forward sampling method* proposed by Fung and Del Favero (1994), the *Markov sampling method* presented by Pearl (1986) and the *systematic sampling method* introduced by Bouckaert (1994b). Other simulation methods could be found in Chavez and Cooper (1990), Dagum and Horvitz (1993), Hryceij (1990), Jensen et al. (1993) and Salmerón et al. (2000).

Despite the huge number of simulation methods, the most intuitive and well-known one is *probabilistic logic sampling* (PLS) Henrion, 1988. PLS – see Figure 2.4– takes advantage of how the Bayesian network defines the probability distribution. The values of each variable are generated in a forward way following their ancestral ordering, i.e. a variable is sampled after all its parents have been sampled.

3

Optimisation

The optimisation task is a central point in several fields. From the point of view of optimisation, a variety of problems could be solved. In fact, the optimisation task is a foundation of the work presented in this dissertation, where the Bayesian network induction is performed as an optimisation task. In this way, some classic optimization algorithms applied in this work to the Bayesian network induction are explained in depth.

This chapter is arranged as follows. First, Section 3.1 presents some basic concepts about the optimisation problem and the terminology used. Then, in Sections 3.2 and 3.3, sequential and population-based optimisation algorithms are respectively introduced.

3.1 Terminology and basic concepts

Optimisation is a crucial topic in fields such as computer science, artificial intelligence or operational research. Optimisation is the process of trying to find the best feasible solution (minimum or maximum) of an optimisation problem. An *optimisation problem* is a problem with a set of possible solutions and an idea of solution quality. Thus, an optimisation problem appears when a set of solutions must be compared to choose the best one depending on its quality.

Formally, an optimisation method tries to solve the following problem:

$$\max_{e \in \mathcal{E}} f(e)$$

such that

$$g_i(e) \leq t_j \quad i = 1, \ldots, n$$
$$h_j(e) = s_j \quad j = 1, \ldots, m$$

that is, to maximise (or minimise) a scoring function in a search space with some constraints. Therefore, an optimisation problem is a pair $(\mathcal{E}, f)$ where $\mathcal{E}$

is a set of feasible solutions (or search space) and f a scoring function which assigns a value to each element of $\mathcal{E}$.

Given an optimisation problem, global and local optima can be defined. A *global optimum* is an element $e_{opt} \in \mathcal{E}$ if $\forall e \in \mathcal{E} \quad f(e_{opt}) \geq f(e)$. In order to define a local optimum, first the definition of neighbourhood is required. The *neighbourhood* of $e \in \mathcal{E}$, $\mathcal{N}(e)$, is a subset of solutions in $\mathcal{E}$ that are close to e. Thus, a *local optimum* is an element $e_{opt} \in \mathcal{E}$ if $\forall e \in \mathcal{N}(e) \quad f(e_{opt}) \geq f(e)$.

Depending on the search space exploration method, an optimisation algorithm can be local or global. *Global optimisation* means that the search engine explores the whole search space at once. In contrast, *local optimisation* locally solves the problem in the neighbourhood of the current solutions. While local optimisation has been thoroughly studied and a high number of numerical methods and algorithms is available, global optimisation is a relatively recent area.

Although this is a good classification for search methods, other systems of classification can be found in the literature. For instance, the optimisation algorithms can be classified as either *deterministic* or *stochastic* methods. A deterministic search algorithm produces the same outcome with the same initial conditions. On the other hand, a stochastic search method produces different final outcomes with the same initial conditions.

Nevertheless, for the purposes of this work, the search algorithms are divided into: *sequential* and *population-based* methods. Depending on the solutions maintained by the search engine during the process, the search algorithms can be defined as either sequential or population-based. When only one solution is changed and maintained during the search process, the search algorithm is considered sequential. However, when a set of solutions is maintained and used to reach the optimum, the search method is defined as population-based.

3.2 Sequential optimisation algorithms

The sequential optimisation methods appeared when artificial intelligence was still a very young field. The first two search techniques are known as *depth first* and *breadth first* methods. These methods are exhaustive, that is, they explore the whole search space and find the global optima. They can be described looking at the *search tree*. A search tree represents all the possible paths, from the starting point to each feasible solution.

Based on the depth first method, *hill climbing* is a graph search algorithm where the current path is extended with a successor node which is closer to the solution than the end of the current path. The closer solution is found in the neighbourhood of the current solution. Contrary to the former algorithms, the hill climbing procedure does not explore the whole search tree. The process stop when a closer node cannot be reached. Similarly, the *best-first* search method is based on the breadth first procedure.

HC
  Select an initial solution $e_0 \in \mathcal{E}$
  while $\exists e \in \mathcal{N}(e_0)\ f(e) \geq f(e_0)$ **do**
    Select e from $\mathcal{N}(e_0)$ in such a way that $f(e) < f(e_0)$
    $e_0 = e$
  end while
  Output e_0 as an optimum

Fig. 3.1. Pseudocode of the hill-climbing algorithm.

The hill climbing algorithm –see Figure 3.1– is a deterministic greedy method: when a decision is taken it cannot be reconsidered. Therefore, when a local optimum is reached, the algorithm stops. In order to overcome this problem, a multi-start hill climbing procedure can be performed. Nevertheless, ideas to avoid the local optima have been developed.

Algorithms like simulated annealing and tabu search are sequential but stochastic. This means that different runs of the algorithm produce different final solutions. *Simulated annealing* Kirkpatrick et al., 1983, inspired by metallurgic annealing, tries to avoid the local optima by accepting small decreases in the scoring function. During the search process the threshold to accept worse solutions is reduced until the simulated annealing becomes a hill-climbing process. However, the number of parameters to be fixed is relatively large when compared with other sequential search methods.

Tabu search Glover and Laguna, 1997 uses a memory of the movements performed to avoid the local optima. Given the current solution, a set of movements in the search space is forbidden. However, at the beginning of the search all movements are allowed. Although the tabu search is a deterministic technique, there are stochastic versions of this method.

In spite of the small number of sequential algorithms presented here, other sequential algorithms to perform the learning of Bayesian network structures are presented in this work. Moreover, the literature of artificial intelligence is full of sequential algorithms such as *beam search* and *branch and bound*.

3.3 Population-based optimisation algorithms

Due to the popularisation of *genetic algorithms* Holland, 1977 by Goldberg (1989), the use of the population-based search process has grown for the last years.

Genetic algorithms try to simulate the natural selection process. The species evolve by means of reproduction and some random changes known as mutation. Genetic algorithms simulate this behaviour to find the optimum. A population of individuals or solutions is maintained and changed during

```
AGA
    Make initial population at random
    while not stop do
        Select parents from the population
        Produce children from the selected parents
        Mutate the individuals
        Extend the population by adding the children to it
        Reduce the extended population
    end while
    Output the best individual found
```

Fig. 3.2. Pseudocode of the abstract genetic algorithm.

the search process. First, the initial population is chosen, and the quality of each of its individuals is determined by means of the scoring function. Next, at every iteration, parents are selected from the population. These parents produce children, which are added to the population. All newly created individuals 'mutate' with a probability near zero, i.e. they change their hereditary distinctions. After that, some individuals are removed from the population according to a selection criterion in order to reduce the population to its initial size. One iteration of the algorithm is referred to as a generation. The pseudocode of an abstract genetic algorithm is shown in Figure 3.2.

The operators which define the child production process and the mutation process are called the crossover operator and the mutation operator respectively. Both operators are applied with different probabilities named crossover probability and mutation probability. Mutation and crossover play different roles in genetic algorithms. Mutation is needed to explore new states and helps the algorithm to avoid local optima. Crossover should increase the average quality of the population.

Most genetic combinatorial approaches have no mechanism to capture the relationships between the domain variables. Genetic algorithms try to implicitly capture these relationships by a semi-blind process, concentrating samples on combinations of high-performance members of the current population through the use of the recombination (crossover and mutation) operators.

In genetic algorithms no explicit information is kept concerning which groups of variables jointly contribute to the quality of candidate solutions. As crossover and mutation operations are randomised, they could break many of these 'desired' relationships between the variables Inza et al., 2001a. Although the search process could produce an individual with an optimal relationship with a subset of variables, a crossover or mutation operator could break this. Therefore, most of the crossover and mutation operations yield unproductive results and the discovery of the global optima could be delayed. On the other hand, genetic algorithms are also criticised in the literature for three aspects

EDA
  $D_0 \leftarrow$ Generate M individuals (the initial population) randomly
  repeat for $l = 1, 2, \ldots$ **until** a stop criterion is met
    $D^s_{l-1} \leftarrow$ Select $N \leq M$ individuals from D_{l-1} according to a selection method
    $p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D^s_{l-1}) \leftarrow$ Estimate the joint probability of selected individuals
    $D_l \leftarrow$ Sample M individuals (the new population) from $p_l(\boldsymbol{x})$

Fig. 3.3. Main scheme of the EDA algorithms.

Larrañaga et al., 2000: (i) the large number of parameters and their optimal selection or tuning process; (ii) the extremely difficult prediction of the movements of the populations within the search space; (iii) their incapacity to solve the well-known deceptive problems.

All these issues related to genetic algorithms create a atmosphere favourable for the appearance of other population-based algorithms. One idea is to estimate the joint distribution of promising solutions and use this estimate in order to generate new individuals. A general scheme of the algorithms based on this principle is called the Estimation of Distribution Algorithm (EDA) Larrañaga and Lozano, 2001; Mühlenbein and Paaß, 1996. In EDAs (see Figure 3.3), there are neither crossover nor mutation operators, and the new population is sampled from a probability distribution which is estimated from a subset of individuals selected from the current population. The initial M individuals are generated at random. These individuals constitute the initial population D_0, and each of them is evaluated by means of the scoring function. At a second step, a number N ($N \leq M$) of individuals is selected. Then, the induction of the n-dimensional probabilistic model that reflects the relationships between the variables is carried out. Finally, M new individuals, which form the new population, are obtained from simulation of the probabilistic distribution learned at the previous step.

The main problem with EDAs is how the probability distribution $p_l(\boldsymbol{x})$ is estimated. Obviously, the computation of all the parameters needed to specify the probability distribution is impractical. This has led to several approximations where the probability distribution is assumed to factorise according to a probability model. For instance, if the problem variables are discrete, approaches such as UMDA Mühlenbein, 1998 and PBIL Baluja, 1994 consider that there are no dependencies between the domain variables. MIMIC De Bonet et al., 1997 and COMIT Baluja and Davies, 1997 take into account relationships between pairs of variables. Finally, FDA Mühlenbein and Mahnig, 1999 and EBNA Etxeberria and Larrañaga, 1999 consider multiple dependencies between the domain variables. Consult Larrañaga and Lozano, 2001 for details.

Several population-based algorithms have been developed in the past years. Nevertheless, only some of them are widely known and applied over real-world problems. Two of them are the ant colony optimisation and the scatter search. The *ant colony optimisation* method Dorigo et al., 1996 is a population-based algorithm inspired in how ant colonies find the shortest path between the food and the ant hill. The *scatter search* Glover, 1998 uses strategies for search diversification and intensification combining already existing solutions to create new solutions.

Part II

Learning Bayesian Networks from Data with Factorisation Purposes

4

Score+search approaches to learning Bayesian network structures

The learning of Bayesian networks from data is becoming a crucial task in the machine learning field. To perform this learning, the literature has proposed a huge number of different methods, the most prominent of which is the score+search approach. In this chapter, the methods to learn Bayesian network provided by the literature are reviewed.

The chapter is organised as follows. Section 4.1 briefly introduces the advantages of learning Bayesian networks from data in an automatic way. In Section 4.2, a revision of the algorithms based on conditional independence tests to learn Bayesian networks is proposed. The score+search approach to automatically learn a Bayesian network is presented in Section 4.3, and the components required to perform the task are analysed. Finally, Section 4.4 briefly reviews the approaches which combine the conditional independence test with the score+search method.

4.1 Introduction

In the last ten years, probabilistic graphical models have become popular as a tool that can structure a set of domain variables. Bayesian networks are accepted probabilistic graphical models used in a large number of domains. When there is no expert to trace the probabilistic relationship of a domain or the elicitation process is too long, the automatic learning of a Bayesian network from a set of examples is a useful and widely accepted alternative.

As it could be seen in Chapter 2, a Bayesian network has two components: a network structure and a set of conditional probabilities. The network structure is a directed acyclic graph which depicts the relationships between the variables or nodes using arrows. The conditional probability measures represent the uncertainty of the domain. As each random discrete variable X_i $(i = 1, \ldots, n)$ follows a multinomial probability distribution, in a Bayesian network the joint probability distribution $p(x_1, \ldots, x_n)$ could be factorised by:

$$p(x_1, \ldots, x_n) = \prod_{i=1}^{n} p(x_i |\ \boldsymbol{pa}(x_i))$$

where x_i represents the value of the random variable X_i, and $\boldsymbol{pa}(x_i)$ represents a combination of the values of the random variable parents of X_i in the directed acyclic graph.

The learning of a Bayesian network from data involves Bayesian network structure induction and parameter learning. Due to it is considered more challenging than parameter learning, this work focusses on the structure induction task.

The automatic learning of a Bayesian network structure reflects the conditional (in)dependences Dawid, 1979 which implicitly appear in the domain database. Although the first automatic approaches have tried to produce a list of conditional (in)dependences by using statistical tests Spirtes et al., 1993, another automatic approach has strongly emerged in the last few years: the score+search approach Buntine, 1996; Cooper and Herskovits, 1992. It is based on the idea of performing an intelligent search in a specific space (the space of network structures, orders, skeletons or equivalence classes), evaluating each proposed Bayesian network, and trying to find the network that best fits the database.

Learning a Bayesian network from data as a score+search approach is proved to be an NP-hard problem when the Bayesian Dirichlet equivalent (BDe) metric is the objective function Chickering et al., 1994. Nevertheless, it is assumed that when other scores are used, the complexity of the problem is not reduced. Moreover, recent works Dasgupta, 1999; Meek, 2001 support this assumption. In this sense, learning a Bayesian network by a score+search procedure could be stated as an NP-hard optimisation problem. This fact justifies the use of heuristic search procedures in order to solve the learning.

4.2 Algorithms based on conditional independence tests

The Bayesian network learning algorithms based on conditional (in)dependence tests are relatively independent of the quantitative representation model of the network. The main objective is not to obtain a network where the probabilistic distribution is similar to the original. Nevertheless, the methods based on conditional independence tests aim to accomplish a study of the (in)dependence relationships among the variables and try to model these relationships into a structure.

The input information for the algorithms based on conditional independence tests could have either of the following forms de Campos, 1998:

- A database from which, with the help of a statistical tests, it is possible to determine the correctness of some conditional (in)dependence relationships.

- An n-dimensional probability distribution to test the feasibility of the conditional (in)dependence relationship.
- A list containing relations of conditional dependence and independence among triplets of variables.

From a formal point of view there are no differences between the three types of input information. However, looking at the required computational time, some differences could be appreciated. These algorithms are based on the structural properties of the model. In general, the best model representation is found when the model is representable in an acyclic directed graph. Nevertheless, when a dataset is used to learn the Bayesian network, they require a high number of cases in order to guarantee the independence test reliability.

In order to learn a network, it is assumed that the results of the (in)dependence test match the independence relationships of the model. Moreover, it is assumed that all the relevant variables of the problem are taken into account and all the cases follow the same relationship when the input information is a database. Using these algorithms, the network construction method is independent of the representation paradigm of the domain knowledge. This way, the algorithms are based on the study of the model structural properties. Thus, when a directed acyclic graph is used to represent the model, the best representation can generally be found Puerta, 2001.

The PC algorithm Spirtes et al., 1991 is the most popular algorithm based on conditional independence tests to learn a Bayesian network. It starts with the complete undirected graph, then it 'thins' the graph by removing edges with zero order conditional independence relations, it 'thins' again with first order conditional independence relations, and so on.

However, the central point of the algorithms based on (in)dependence test is the number and complexity of the tests carried out producing an effect on the efficiency of the algorithms. Recent works try to reduce the complexity. For instance, Cheng et al. (2002) only requires a polynomial number of conditional independence tests in typical cases and de Campos and Huete (2000) makes an intensive use of low order conditional independence tests. The research in this area proves that under conditions of complete data and given a node ordering, conditional independence tests for learning Bayesian networks from data are equivalent to local scoring metrics Cowell, 2001.

4.3 Algorithms based on scoring functions

The assumption that the complexity of the problem is NP-hard is supported by several papers in the literature of the field Chickering et al., 1994; Dasgupta, 1999; Meek, 2001. This issue is widely accepted in the community. It motivated the use of algorithm based on scoring function to learn Bayesian networks. Thus, from this point of view, the learning of Bayesian networks

is an optimisation problem where three components should be taken into account:

- A *scoring function* in order to assess the quality of the network and select the best solution.
- A *search space* to influence the number of feasible solutions.
- A *search engine* to determine the strategy of the search.

These components are analysed in the next sections.

4.3.1 Families of scoring functions

In order to select the network with the highest score, a scoring metric $g(S, D)$ allows to sort a set of Bayesian networks given a dataset D. Hence, the aim is to attain a Bayesian network with a high scoring value.

A good scoring measure should observe a set of characteristics. The most interesting characteristic is that when two different networks reflect the same set of (in)dependence relationships among the variables, the scoring measure should assigns the same value. Other desirable characteristics are as follows:

- The perfect representations should have higher scoring values than the other representations.
- The minimal I-map should have a higher scoring value than the non-minimal I-map.
- The networks reflecting information of dataset D should have a higher scoring value than the networks that do not reflect this information.

The literature Bouckaert, 1995; Buntine, 1991; Heckerman et al., 1995 provides a deeper analysis of the characteristics of scoring functions.

Penalised maximum likelihood

Given a dataset D with N cases, the *maximum likelihood* estimate, $\widehat{\boldsymbol{\theta}}$, for the parameters $\boldsymbol{\theta}$ and the associated maximised likelihood, $p(D|S, \widehat{\boldsymbol{\theta}})$, could be calculated for any structure S. Thus, the likelihood could be applied as a scoring measure. For computational reasons, the likelihood is usually calculated applying the logarithm. Hence, it is defined as:

$$\begin{aligned} LL(D|S) &= \log p(D|S, \boldsymbol{\theta}) = \log \prod_{l=1}^{N} p(\boldsymbol{x}_l|S, \boldsymbol{\theta}) \\ &= \log \prod_{l=1}^{N} \prod_{i=1}^{n} p(x_{l,i}|\boldsymbol{pa}_i^S, \boldsymbol{\theta}_i) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} \log(\theta_{ijk})^{N_{ijk}} \end{aligned}$$

where N_{ijk} is the number of cases in D where the variable X_i has the x_i^k value and $\boldsymbol{pa}_i$ has its j^{th} value.

Using the maximum likelihood estimate for θ_{ijk}, given by $\widehat{\theta}_{ijk} = \frac{N_{ijk}}{N_{ij}}$, the formulation is as follows:

$$LL(D|S) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}}$$

where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

When the model is complex, the sampling error related to the maximum likelihood estimator implies that the maximum likelihood estimate is not actually a believable value for the parameter. On the other hand, the monotonicity of the likelihood with respect to the complexity of the structure usually leads the search through complete networks. Therefore, in order to overcome these difficulties, a penalisation term is added. When some form of penalty model complexity is included into the maximised likelihood, a general formula for the *penalised maximum likelihood* score is obtained:

$$\sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - f(N) dim(S)$$

where $dim(S)$ is the number of parameters needed to specify the model of the Bayesian network with a structure given by S, and $f(N)$ is a non-negative penalisation function. Usually, $dim(S) = \sum_{i=1}^{n} q_i(r_i - 1)$.

A usual choice for the penalty function is the Jeffreys-Schwarz criterion, generally called the Bayesian Information Criterion (BIC) Schwarz, 1978, where $f(N) = \frac{1}{2} \log N$. Here, the penalised maximum likelihood is as follows:

$$\sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - \left(\frac{1}{2} \log N \right) \sum_{i=1}^{n} q_i(r_i - 1)$$

Another well-known example is Akaike's Information Criterion (AIC) Akaike, 1974 where $f(N) = 1$.

Bayesian scores. Marginal likelihood

In the Bayesian approach to Bayesian network model induction from data, the uncertainty about the model (structure and parameters) is expressed by defining a variable whose states correspond to the possible network structure hypothesis S^h, and by assessing the probability $p(S^h)$. Then, given a random sample D from the probability distribution $\boldsymbol{X}$, the posterior distribution of the structure given the dataset $p(S^h|D)$ and the posterior distribution of the parameters given the structure and the dataset $p(\boldsymbol{\theta}_S|D, S^h)$ are calculated.

Using the Bayes rule, $p(S^h|D)$ and $p(\boldsymbol{\theta}_S|D, S^h)$ are as follows:

$$p(S^h|D) = \frac{p(S^h)p(D|S^h)}{\sum_S p(S)p(D|S)}$$
$$p(\boldsymbol{\theta}_S|D, S^h) = \frac{p(\boldsymbol{\theta}_S|S^h)p(D|\boldsymbol{\theta}_S, S^h)}{p(D|S^h)}$$

where $p(D|S^h) = \int p(D|\boldsymbol{\theta}_S, S^h)p(\boldsymbol{\theta}_S|D, S)d\boldsymbol{\theta}_S$.

In the *Bayesian model averaging* approach, the joint distribution for $\boldsymbol{X}$, $p(\boldsymbol{x})$, is estimated by averaging over all possible models and their parameters:

$$p(\boldsymbol{x}) = \sum_S p(S|D) \int p(\boldsymbol{x}|\boldsymbol{\theta}_S, S)p(\boldsymbol{\theta}_S|D, S)d\boldsymbol{\theta}_S$$

In the approach known as *Bayesian model selection*, the selected model is the one whose logarithm of relative posterior probability, $\log p(S, D)$, is maximum. Taking this

$$\log p(S \mid D) \propto \log p(S, D) = \log p(S) + \log p(D \mid S)$$

into account, and assuming that the prior distribution over the structure is uniform, an equivalent criterion is the log *marginal likelihood*, $\log p(D \mid S)$, of the data given the structure. It is possible to compute the marginal likelihood efficiently and in a closed form under a number of general assumptions Cooper and Herskovits, 1992; Heckerman et al., 1995. For instance, Cooper and Herskovits (1992) proves that if the cases occur independently, if there are no missing values, and if the prior probability density functions of the parameters given the structure are uniform, then:

$$p(D \mid S) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!.$$

This score is known as the K2 metric. A number of modifications of the K2 score metric were introduced in Borgelt and Kruse (2001) in order to control the tendency towards simpler network structures.

However, the K2 metric could be criticised for assuming a uniform prior probability density over the model parameters. This leads Heckerman et al. (1995) to derive the *Bayesian Dirichlet* (BD) metric. A generalisation of the K2 score is proposed, allowing the prior probability density functions for the Bayesian network parameters given the structure to be Dirichlet distributions. Therefore, under some assumptions:

$$p(D|S) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}$$

where $\Gamma(\cdot)$ is the *gamma function* and $\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$.

In addition to the BD metric, the *Bayesian Dirichlet equivalent* metric Heckerman et al., 1995 is presented. This metric assumes the *likelihood*

equivalence stating that the dataset should not help to discriminate between Bayesian network structures that represent the same set of conditional independences.

Information theory scores

Scores that compare two joint probability distributions are called *scoring rules*. $S(p(\boldsymbol{x}), p'(\boldsymbol{x}))$ denotes the scoring rule used to compare the true joint probability distribution $p(\boldsymbol{x})$ and the one approximated by the Bayesian network evaluated $p'(\boldsymbol{x})$. A score is called a *proper scoring rule* if $S(p(\boldsymbol{x}), p(\boldsymbol{x})) \geq S(p(\boldsymbol{x}), p'(\boldsymbol{x}))$ for all $p'(\boldsymbol{x})$. Although there is an infinite number of functions that could be applied as a proper score McCarthy, 1956, the *logarithmic score* has received special attention in the literature:

$$S(p(\boldsymbol{x}), p'(\boldsymbol{x})) = \sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log p'(\boldsymbol{x})$$

The logarithmic scoring rule verifies the interesting property of being equivalent to the *Kullback Leibler cross-entropy measure* Kullback and Leibler, 1951:

$$\begin{aligned} D_{K-L}(p(\boldsymbol{x}), p'(\boldsymbol{x})) &= \sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{p'(\boldsymbol{x})} \\ &= \sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log p(\boldsymbol{x}) - \sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log p'(\boldsymbol{x}) \end{aligned}$$

This formula represents the difference between the information contained in the true joint probability distribution $p(\boldsymbol{x})$ and the information contained in the approximate joint probability distribution $p'(\boldsymbol{x})$. Since the expression $\sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log p(\boldsymbol{x})$ does not depend on the approximate joint probability distributions, the logarithmic scoring rule is a linear transformation of the Kullback Leibler cross-entropy measure. Therefore, minimising the Kullback Leibler cross-entropy measure is equivalent to maximising the logarithmic scoring rule.

The *minimum description length* (MDL) principle Rissanen, 1978 is related to this approach. The MDL principle states that the best model structure for a database is the one that minimises the sum of encoding lengths for the database and the model. It can be said that this approach to learning Bayesian network structures belongs to the penalised maximum likelihood score class, as well as to the Bayesian approach Lam and Bacchus, 1994. Nevertheless, the MDL development could be seen as a generalisation of the Kullback Leibler cross-entropy measure.

When applying the MDL principle to learn Bayesian network structures, the description length of the database D is:

$$\sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} -N_{ijk} \log \frac{N_{ijk}}{N_{ij}}$$

and the required length to store the parameters is given by $\frac{1}{2}\log Ndim(S)$. As this length is constant for all the Bayesian network structures with n variables, the structure of the Bayesian network is not encoded. Thus, the MDL score is as follows:

$$\sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}\log\frac{N_{ijk}}{N_{ij}} - \frac{1}{2}\log Ndim(S).$$

It must be noted that the MDL equals the BIC score in spite of the differences within the derivation process.

The literature includes several scores where the information theory is taken into account to learn Bayesian network structures. Works such as Herskovits and Cooper (1990) and Geiger (1992) propose the use of the *entropy* of a probability distribution represented by a Bayesian network to induce model structures. It is shown that the entropy of the distribution represented by a Bayesian network structure S is:

$$H_S = \sum_{i=1}^{n}\sum_{j=1}^{q_i} p(\mathbf{pa}_i = j) H_{X_i|\mathbf{pa}_i=j}$$

where $H_{X_i|\mathbf{Pa}_i=j} = \sum_{k=1}^{r_i} p(X_i = x_i^k \mid \mathbf{pa}_i = j)\ln p(X_i = x_i^k \mid \mathbf{pa}_i = j)$.

4.3.2 The search space

The nature search space, as the collection of feasible solutions to a problem, influences the search process. However, due to the intrinsic characteristics of the search space, such as the required computational cost to compute a neighbour solution or the number of solutions included, the search processes are not always performed over the 'best' search space.

In the learning of Bayesian network structures, the most usual approach to perform the search of the Bayesian network model is to implement it in the *space of directed acyclic graphs*. In spite of the fact that the space of directed acyclic graphs grows more than exponentially in the number of variables Robinson, 1977, this search space is widely used. Moreover, the search process in this space has difficulties, Anderson et al. (1997) reports the three main difficulties when performing the search in the space of directed acyclic graphs:

- Computational inefficiency related to repeating analyses for equivalent directed acyclic graphs.
- Severe constraints on prior distributions in order to ensure that equivalent directed acyclic graphs have equal posterior probabilities.
- Bayesian model averaging methods require an explicit representation of the equivalence classes to achieve specified weights for these classes by weighting individual directed acyclic graphs.

Nevertheless, Chickering (1995) reports that AIC, BIC, MDL and BDe metrics assess two Bayesian networks with the same set of conditional independences have the same scoring measure value. In this way, the search could also be performed in the *space of equivalence classes* (classes that reflect the same set of conditional independences). An equivalence class of Bayesian network structures represents all the directed acyclic graphs that encode the same set of conditional independences. Each equivalence class has a canonical representation Anderson et al., 1997; Chickering, 1995; Spirtes et al., 1993 in the form of an acyclic partially directed graph where the edges could be directed and undirected and satisfy characterising conditions. Although the space of equivalence classes has been used to learn Bayesian networks Chickering, 1995, recent works Gillispie and Perlman, 2001 note the relationship between the cardinality of Bayesian network structure space and equivalence of class space: this could be interpreted as a deceleration in the popularisation of this promising approach. There are two basic reasons for this deceleration: the cardinality of this space is not largely reduced and the search process in this space has a large computational cost. However, with a convenient graphical representation for the equivalence class structure and a set of suitable operators to move in the space, a score+search framework in this space slightly outperforms the results of a traditional approach Chickering, 2002.

Although the search of Bayesian network structures is usually performed in a single search space, literature reports attempt to combine the characteristics of both search spaces: directed acyclic graphs and equivalence classes Kočka and Castelo, 2001.

The literature also proposes to perform the search in the *space of skeletons* Steck, 2000; van Dijk et al., 2003. The advantage of this space for heuristics inspired by evolutionary computation is that the operations performed with the old population to create a new one are closed: valid individuals are always generated in the offspring without the need of repair-operators. However, it implies the necessity of finding the orientation of the edges.

Other authors Friedman and Koller, 2003; Larrañaga et al., 1996a; Puerta, 2001 propose to perform the search in the *space of orders* of the n variables of the problem. The motivation for the birth of this approach is that several structure learning algorithms need the n variables to be ordered as well as a smoother posterior landscape (a small number of local optima and similar orders obtain similar scoring measures).

4.3.3 The search engine

The assumption that learning Bayesian networks from data is an NP-hard problem justifies the development of methods to face the problem. For years, algorithms which tackle the induction of Bayesian networks have been proposed. Although the search engines proposed in the literature could be divided into different axes, they could be classified in two basic families: *sequential* and *population-based* algorithms.

B

Step 1: Calculate for each pair of variables $A[X_i, X_j] = g(X_i, X_j) - g(X_i, \emptyset)$
Step 2: Repeat until $A[X_i, X_j] \leq 0$ for each pair of variables or $A[X_i, X_j] = -\infty$
  Step 2.1: Select the pair of variables such as $(X_s, X_r) = \max_{s,r} A[X_s, X_r]$
  Step 2.2: If $A[X_s, X_r] > 0$ then $\boldsymbol{pa}_s = \boldsymbol{pa}_s \cup \{X_r\}$
  Step 2.3: For all $X_l \in \boldsymbol{pa}_s$ and $X_k \in \boldsymbol{des}_s \cup \{X_s\}$, set $A[X_l, X_k] = -\infty$
  Step 2.4: Set $A[X_s, X_r] = -\infty$
  Step 2.5: Recalculate the score of all the feasible parents of X_s as follows:
    Step 2.5.1: if $A[X_s, X_k] = -\infty$ then set $A[X_s, X_k] = g(X_s, \boldsymbol{pa}_s \cup \{X_k\}) - g(X_s, \boldsymbol{pa}_s)$

Fig. 4.1. Pseudocode of the B algorithm Buntine, 1991.

K2

Repeat for each variable $X_i \in S$ following the input ordering
Step 1: Let $\boldsymbol{pa}_i = \emptyset$ and $p_{old} = g(X_i, \boldsymbol{pa}_i)$. Set the boolean $noStop = true$
Step 2: Repeat while $noStop$ and $|\boldsymbol{pa}_i| < maxParents$
  Step 2.1: Let X_z be the node in $pred(X_i) \setminus \boldsymbol{pa}_i$ which maximises $g(X_i, \boldsymbol{pa}_i \cup \{Z\})$
  Step 2.2: Set $p_{new} = g(X_i, \boldsymbol{pa}_i \cup \{X_z\})$
  Step 2.3: If $p_{new} > p_{old}$ then let $p_{old} = p_{new}$ and $\boldsymbol{pa}_i = \boldsymbol{pa}_i \cup \{X_z\}$; else $noStop = false$

Fig. 4.2. Pseudocode of the K2 algorithm Cooper and Herskovits, 1992.

When a single solution is recursively built, or a unique solution is kept during the optimisation process, the search method could be regarded as sequential. Among sequential algorithms, the hill-climbing algorithm (also known as the B algorithm –see Figure 4.1–) Buntine, 1991 is the most popular sequential procedure to perform the search in the space of Bayesian network structures. It is a greedy algorithm which adds the arc that maximises the scoring in each iteration starting from an empty solution. The B algorithm only takes into account one operator in the search space: the arc addition. However, variations of this algorithm could be presented including several operators. For instance, in this work, the B3 algorithm performs a greedy search taking into account three operators: arc addition, arc deletion and arc reversal.

The well-known K2 algorithm Cooper and Herskovits, 1992 is a variation of the hill-climbing search procedure. Figure 4.2 shows its pseudocode. The K2 requires two parameters: a domain variable order and an upper bound of the number of parents of the variables. K2 looks for the best set of parents of each variable within the previous subset of variables. However, it requires a de-

composable scoring measure, i.e, $g(S) = \sum_{i=1}^{n} g(X_i, \boldsymbol{pa}_i)$. Another variation of the hill-climbing method is the iterated hill-climbing approach Chickering et al., 1995. This procedure is motivated by the strong dependence of the hill-climbing method on the starting point. It proposes to start the search in several random points of the search space and it returns the best solution found so far. Other greedy methods to learn Bayesian networks can be found in Bouckaert, 1995; Chickering, 2002; Friedman, 1997; Pazzani, 1996.

The classic sequential greedy search methods suffer from the 'nesting' problem: when a decision is taken, it cannot be reconsidered. For instance, in the B algorithm, when an arc is added to the solution, it cannot be deleted later. Avoiding this problem is a crucial task in order to escape from the local optima and attain a better structure. The floating search methods have been adapted to learn Bayesian networks from data Blanco et al., 2002; Blanco et al., 2004a –see Section 5.2 for details–. The number and type of the decision (arc addition or arc deletion) are taken in runtime depending on the characteristics of the problem and the value of the score function. Other 'semi' hill-climbing approaches to learn Bayesian network from data have been proposed in the literature: branch-and-bound Tian, 2000 or beam search Friedman et al., 1999, for instance.

All previous sequential algorithms are *deterministic* methods, i.e. the same initial conditions produce the same final solution. *Stochastic* algorithms, on the other hand, produce different finals solutions with the same initial conditions. Therefore, sequential stochastic algorithms have been proposed to learn Bayesian networks. A variable neighbourhood search, also proposed as a modification of the hill-climbing procedure de Campos and Puerta, 2001, uses a systematic change of neighbourhood space within a randomised local search algorithm. General purpose stochastic procedures have been presented in the literature for Bayesian network induction: simulated annealing Bouckaert, 1995; Chickering et al., 1995 or tabu search Bouckaert, 1995; Munteanu and Cau, 2000.

The algorithms which relax the constraints of the classic greedy search could be seen as extensions or approximations of the hill-climbing approach. A different point of view to perform the search of a Bayesian network structure in a sequential way is given by the Markov Chain Monte Carlo (MCMC) Friedman and Koller, 2003 algorithm. It constructs a Markov chain over orders. To tackle the problem of learning Bayesian networks, it samples entire networks from the posterior probability given the order.

In opposition to sequential methods, population-based approaches maintain a set of solutions and combine them to obtain a new set of solutions. Then, the best solution found during the search process is considered the output solution. All population-based methods are stochastic algorithms. Within this category, several methods appear in the literature. The genetic algorithms Larrañaga et al., 1996a; Larrañaga et al., 1996b; Myers et al., 1999, evolutionary programming Wong et al., 1999, ant colony optimisation de Campos et al., 2002 and estimation of distribution algorithms Blanco et al., 2003; Romero

et al., 2004 have been presented for Bayesian network induction. Nevertheless, they have to tackle the possible non-validity of new individuals. Hence, the widely used solution is the inclusion of a 'repairing' procedure which converts the non-valid individuals into valid. This fact is motivated by the non-closure with respect to the search space of the majority of the basic operators used. Novel operators to preserve the closure are proposed in the literature Cotta and Muruzábal, 2002.

4.4 Hybrid algorithms

Once the algorithms based on conditional independence tests and on scoring functions are described, the existence of hybrid algorithms must be noted. Hybrid methods do not fall clearly in any of the two categories presented. Although they perform a search process using a scoring measure, they also explicitly use the conditional independences embodied in Bayesian networks to calculate the scoring function, carrying out independence test to limit the search.

The first attempts to take advantage of the 'best' of each type of algorithms are reported in Singh and Valorta (1993) where the K2 algorithm is inserted into a PC algorithm with the aim of overcoming the ordering required. Acid and de Campos (1996) presents the *BElief NEtworks DIscovery using Cut-set Techniques* (BENEDICT) algorithm, which constructs a Bayesian network in a greedy way performing conditional independence tests to calculated the scoring measure. Interesting studies of hybrid algorithms may be found in Acid, 1999; Puerta, 2001.

5

New methods to learn Bayesian network structures

A set of algorithms for Bayesian network structure induction is presented in this chapter. Within the score+search approach to learning Bayesian network structures, the floating methods, the greedy randomized adaptive search procedure and the EDAs are introduced.

The chapter is arranged as follows. Section 5.1 briefly introduces the use of heuristic search procedures to learn Bayesian networks. The original floating methods and the corresponding adaptation to the problem are proposed in Section 5.2, together with the presentation of the experimental results obtained by these algorithms. In Section 5.3, the greedy randomised adaptive search procedure and the respective experimental results are introduced. The adaptation of the EDAs to learn Bayesian networks is presented in Section 5.4, where some promising results of this approach are also shown.

This work is an extension and adaptation of Blanco et al. (2004a), Blanco et al. (2003) and Blanco et al. (2002).

5.1 Introduction

The algorithms that perform the learning of Bayesian networks from data could be divided into two main groups: algorithms based on conditional independence tests and algorithms based on a scoring function (or score+search approach). Both types of approaches are explained in Chapter 4.

However, the machine learning community has generally accepted that learning Bayesian networks from data is NP-hard. A number of complexity results Bouckaert, 1994a; Chickering et al., 2003; Chickering, 1996; Chickering et al., 1994; Dasgupta, 1999; Meek, 2001 have emerged over the last years which indicate that this belief is well founded. Therefore, the heuristics search procedures to learn Bayesian network as an optimisation problem have experimented a growth during the last years –see Section 4.3 for details–.

In this work, some heuristic procedures are proposed to perform the search of the 'best' Bayesian network structure. The search is carried out over the

space of directed acyclic graphs. The usual representation (applied in all the experiments of this chapter) of a network structure is a connectivity matrix. In an n-dimensional domain, each Bayesian network structure is represented by a connectivity matrix $C \in M(n,n)$, whose elements c_{ij} verify that:

$$c_{ij} = \begin{cases} 1 \text{ if } X_i \text{ is parent of } X_j \\ 0 \text{ otherwise.} \end{cases}$$

5.2 Floating methods to learn Bayesian network structures

Floating search methods appeared some years ago to solve the feature selection problem. In *feature subset selection* –see Section 7.2 for details–, the main goal is to obtain a subset of domain variables. This subset is formed by the 'best' domain variables with respect to a score metric. In pattern recognition, the number of variables to be selected is usually previously fixed as a parameter m, and the method stops when m variables from n ($m < n$) are selected.

The first algorithms proposed for feature selection were *sequential forward selection* (SFS) and *sequential backward elimination* (SBE). These methods are classic hill-climbing greedy algorithms. Nevertheless, both algorithms suffer from the 'nesting' problem: when a decision is made, it cannot be reconsidered. In this case, when a feature is added (or removed), it cannot be discarded (or considered) later. To avoid this effect, other sequential but more sophisticated algorithms were proposed, such as *step-wise* Wahba, 1988 and *l-r* Stearns, 1976. At each iteration, the *step-wise* algorithm decides whether to add or to remove a single variable, depending on the best score metric achieved. The *l-r* method is similar to *step-wise*, but it considers whether to add l variables or to remove r variables. Although these algorithms try to partly avoid the nesting problem, there is no way to determine the best values of l and r in the case of the *l-r* algorithm. The procedure proposed by *sequential forward floating selection* (SFFS) and *sequential backward floating selection* (SBFS) Pudil et al., 1994 allows l and r values to intelligently 'float'. They do no have to be fixed previously.

5.2.1 The original floating methods

Before explaining the original algorithms, a number of preliminaries are required. Bearing in mind the feature selection problem, let $\mathcal{X}_k = \{X_i \in \mathcal{Y}; 1 \leq i \leq k\}$ be a subset of domain variables, where $\mathcal{Y}$ is the total set of domain variables, and let $J(X_i)$ be the value of the objective function, only for the X_i variable. Thus, the *individual significance* of the variable X_i is defined as $J(X_i)$. Obviously, $J(\mathcal{X}_k)$ is the value of the $\mathcal{X}_k$ subset of variables.

Then, the *significance* $S_{k-1}(X_j)$ *of the variable* $X_j \in \mathcal{X}_k$ is defined as:

SFFS

Step 1: Inclusion

Using the basic greedy method, select feature X_{k+1} from the set of available measurements, $\mathcal{Y} \setminus \mathcal{X}_k$, to form feature set $\mathcal{X}_{k+1}$, i.e. add to set $\mathcal{X}_k$ the most significant feature X_{k+1} with respect to $\mathcal{X}_k$. Therefore, $\mathcal{X}_{k+1} = \mathcal{X}_k \cup X_{k+1}$.

Step 2: Conditional exclusion

Find the least significant feature in set $\mathcal{X}_{k+1}$. If X_{k+1} is the least significant feature in set $\mathcal{X}_{k+1}$, i.e. $J(\mathcal{X}_{k+1} \setminus X_{k+1}) \geq J(\mathcal{X}_{k+1} \setminus X_j)\ \forall j = 1, 2, \ldots, k$, then set $k = k + 1$ and return to *Step 1*, but if $X_r\ 1 \leq r \leq k$, is the least significant feature in set $\mathcal{X}_{k+1}$, i.e. $J(\mathcal{X}_{k+1} \setminus X_r) > J(\mathcal{X}_k)$, then exclude X_r from $\mathcal{X}_{k+1}$ to form a new feature set $\mathcal{X}'_k$, i.e $\mathcal{X}'_k = \mathcal{X}_{k+1} \setminus X_r$. Note that now $J(\mathcal{X}'_k) > J(\mathcal{X}_k)$. If $k = 2$, then set $\mathcal{X}_k = \mathcal{X}'_k$ and $J(\mathcal{X}_k) = J(\mathcal{X}'_k)$ and return to *Step 1*. If not, go to *Step 3*.

Step 3: Continuation of conditional exclusion

Find the least significant feature X_s in set $\mathcal{X}'_k$. If $J(\mathcal{X}'_k \setminus X_s) \leq J(\mathcal{X}_{k-1})$, then set $\mathcal{X}_k = \mathcal{X}'_k$, $J(\mathcal{X}_k) = J(\mathcal{X}'_k)$ and return to *Step 1*. If $J(\mathcal{X}'_k \setminus X_s) > J(\mathcal{X}_{k-1})$, then exclude X_s form $\mathcal{X}'_k$ to from a newly reduced set $\mathcal{X}'_{k-1}$, i.e. $\mathcal{X}'_{k-1} = \mathcal{X}'_k \setminus X_s$. Set $k = k - 1$. Now if $k = 2$, then set $\mathcal{X}_k = \mathcal{X}'_k$ and $J(\mathcal{X}_k) = J(\mathcal{X}'_k)$ and return to *Step 1*. If not, repeat *Step 3*.

Fig. 5.1. Original sequential forward floating selection algorithm.

$$S_{k-1}(X_j) = J(\mathcal{X}_k) - J(\mathcal{X}_k \setminus X_j)$$

where $X_j \in \mathcal{X}_k$ is a candidate variable to be removed from $\mathcal{X}_k$. The significance $S_{k-1}(X_j)$ measures the difference between the current subset of variables $\mathcal{X}_k$ and the current subset of variables without the node candidate to be removed $X_j \in \mathcal{X}_k$.

The *significance* $S_{k+1}(X_j)$ *of the variable* $X_j \in \mathcal{Y} \setminus \mathcal{X}_k$ is defined as:

$$S_{k+1}(X_j) = J(\mathcal{X}_k \cup X_j) - J(\mathcal{X}_k)$$

where $X_j \in \mathcal{Y} \setminus \mathcal{X}_k$ is a candidate variable to be included in $\mathcal{X}_k$. The significance $S_{k+1}(X_j)$ measures the difference between the current subset with the candidate variable added $\mathcal{X}_k \cup X_j$ and the current subset $\mathcal{X}_k$.

Figure 5.1 shows the original SFFS algorithm as proposed in Pudil et al. (1994) for feature subset selection. It is assumed that a previous subset of k variables is selected before the process begins. Usually $k = 2$ variables are selected in a greedy way by means of a hill-climbing method. The SFFS starts with an *inclusion* step. In this stage, a candidate feature, to be part of the solution, is selected by a hill-climbing method and added to $\mathcal{X}_k$ obtaining $\mathcal{X}_{k+1}$, and the SFFS goes to the exclusion step. At the *exclusion* step, it is proposed that the least significant feature from $\mathcal{X}_{k+1}$ be removed from the solution. If the least significant feature is the last one inserted, the deletion

SBFS

Step 1: Exclusion

Using the basic greedy method, remove feature X_{k+1} from the set of available measurements, $\mathcal{X}_k$, i.e. delete the least significant feature X_{k+1} from set $\mathcal{X}_k$ to form $\mathcal{X}_{k+1}$.

Step 2: Conditional inclusion

Find the most significant feature in the set of excluded features $\mathcal{Y} \setminus \mathcal{X}_{k+1}$. If X_{k+1} is the most significant feature in the set $\mathcal{X}_{k+1}$, i.e. $J(\mathcal{X}_{k+1} \cup X_{k+1}) \geq J(\mathcal{X}_{k+1} \cup X_j) \forall j = 1, 2, \ldots, k$, then set $k = k + 1$ and return to *Step 1*, but if X_r $1 \leq r \leq k$ is the most significant feature in set $\mathcal{X}_{k+1}$, i.e. $J(\mathcal{X}_{k+1} \setminus X_r) > J(\mathcal{X}_k)$, then include X_r to $\mathcal{X}_{k+1}$ to form a new feature set $\mathcal{X}'_k$, i.e. $\mathcal{X}'_k = \mathcal{X}_{k+1} \cup X_r$. Note that now $J(\mathcal{X}'_k) > J(\mathcal{X}_k)$. If $k = 2$, then set $\mathcal{X}_k = \mathcal{X}'_k$ and $J(\mathcal{X}_k) = J(\mathcal{X}'_k)$ and return to *Step 1*. If not, go to *Step 3*.

Step 3: Continuation of conditional inclusion

Find the most significant feature X_s in the set of excluded features $\mathcal{Y} \setminus \mathcal{X}'_k$. If $J(\mathcal{X}'_k \cup X_s) \leq J(\mathcal{X}_{k-1})$, then set $\mathcal{X}_k = \mathcal{X}'_k$, $J(\mathcal{X}_k) = J(\mathcal{X}'_k)$ and return to *Step 1*. If $J(\mathcal{X}'_k \cup X_s) > J(\mathcal{X}_{k-1})$, then include X_s to set $\mathcal{X}'_k$ to form a newly extended set $\mathcal{X}'_{k-1}$, i.e. $\mathcal{X}'_{k-1} = \mathcal{X}'_k \cup X_s$. Set $k = k - 1$. Now if $k = 2$, then set $\mathcal{X}_k = \mathcal{X}'_k$ and $J(\mathcal{X}_k) = J(\mathcal{X}'_k)$ and return to *Step 1*. If not, repeat *Step 3*.

Fig. 5.2. Original sequential backward floating selection algorithm.

is rejected and the algorithm returns to the inclusion step. If not, the least significant variable from $\mathcal{X}_{k+1}$ is deleted to form $\mathcal{X}'_k$ and the algorithm goes to the *continuation of the conditional exclusion* step. At the continuation of the conditional exclusion step, the algorithm asks for the least significant variable of $\mathcal{X}'_k$. If the value of the scoring function of $\mathcal{X}'_k$ without the least significant feature is less than or equal to the value of the scoring function of $\mathcal{X}_{k-1}$ (the last subset of variables with the same cardinality), it rejects the deletion and returns to the *inclusion* step. If not, the variable is excluded from $\mathcal{X}'_k$ to form $\mathcal{X}'_{k-1}$ and the methods repeats the *exclusion* step.

Note that the inclusion of variables is carried out while the least significant variable in the set is the last one added. The exclusion of variables is performed while the objective function is higher than the objective function of the last subset with the same cardinality. Due to this fact, the number of included (excluded) variables 'float' during the search process.

In Figure 5.2, the corresponding 'up-bottom' floating approach to feature subset selection is depicted. Note that all the steps are similar and opposed to each other. When the SFFS includes (excludes) a feature in the candidate subset, the SBFS procedure proposes the exclusion (inclusion) of a feature in the candidate subset.

Adapted SFFS

Step 1: Inclusion

Using the basic greedy method, select arc a_{ij}^{k+1} from set $\mathcal{A} \setminus \mathcal{E}_k$ to form structure $\mathcal{E}_{k+1}$, i.e. add to structure $\mathcal{E}_k$ the most significant arc a_{ij}^{k+1} with respect to $\mathcal{E}_k$. Therefore, $\mathcal{E}_{k+1} = \mathcal{E}_k \cup a_{ij}^{k+1}$. If it is not possible to include an arc, stop.

Step 2: Conditional exclusion

Find the least significant arc in set $\mathcal{E}_{k+1}$. If a_{ij}^{k+1} is the least significant arc of set $\mathcal{E}_{k+1}$, i.e. $g(\mathcal{E}_{k+1} \setminus a_{ij}^{k+1}) \geq g(\mathcal{E}_{k+1} \setminus a_{ij}^{l}) \forall l = 1, 2, \ldots, k$, then set $k = k + 1$ and return to *Step 1*, but if a_{ij}^{r} $1 \leq r \leq k$ is the least significant arc in set $\mathcal{E}_{k+1}$, i.e. $g(\mathcal{E}_{k+1} \setminus a_{ij}^{r}) > g(\mathcal{E}_k)$, then exclude a_{ij}^{r} from $\mathcal{E}_{k+1}$ to form a new graph $\mathcal{E}'_k$, i.e. $\mathcal{E}'_k = \mathcal{E}_{k+1} \setminus a_{ij}^{r}$. Note that now $g(\mathcal{E}'_k) > g(\mathcal{E}_k)$. If $k = 2$, then set $\mathcal{E}_k = \mathcal{E}'_k$ and $g(\mathcal{E}_k) = g(\mathcal{E}'_k)$ and return to *Step 1*. If not, go to *Step 3*.

Step 3: Continuation of conditional exclusion

Find the least significant arc a_{ij}^{s} in set $\mathcal{E}'_k$. If $g(\mathcal{E}'_k \setminus a_{ij}^{s}) \leq g(\mathcal{E}_{k-1})$, then set $\mathcal{E}_k = \mathcal{E}'_k$, $g(\mathcal{E}_k) = g(\mathcal{E}'_k)$ and return to *Step 1*. If $g(\mathcal{E}'_k \setminus a_{ij}^{s}) > g(\mathcal{E}_{k-1})$, then exclude a_{ij}^{s} from $\mathcal{E}'_k$ to form a newly reduced graph $\mathcal{E}'_{k-1}$, i.e. $\mathcal{E}'_{k-1} = \mathcal{E}'_k \setminus a_{ij}^{s}$. Set $k = k - 1$. Now if $k = 2$, then set $\mathcal{E}_k = \mathcal{E}'_k$ and $g(\mathcal{E}_k) = g(\mathcal{E}'_k)$ and return to *Step 1*. If not, repeat *Step 3*.

Fig. 5.3. Adapted sequential forward floating selection algorithm.

5.2.2 Adaptation of floating methods to learn Bayesian network structures

Once the floating methods for feature selection are presented, the adaptation to learning Bayesian networks is proposed. In this problem, a structure is a graph $\mathcal{G}_k = \{\mathcal{V}, \mathcal{E}_k\}$ where $\mathcal{V} = \{X_1, X_2, \ldots, X_n\}$ is the set of nodes and $\mathcal{E}_k \subset \mathcal{A} = \{a_{11}, a_{12}, \ldots, a_{ij}, \ldots, a_{nn}\}$ denotes the set of arcs. $\mathcal{A}$ denotes the set of all possible arcs between the nodes of $\mathcal{V}$.

Let $g(\mathcal{E}_k)$ be the score metric for the complete structure $\mathcal{E}_k$. Now, the *significance* $S_{k-1}(a_{ij}^{k})$ *of the arc* $a_{ij}^{k} \in \mathcal{E}_k$ between nodes X_i and X_j, that is, the arc candidate to be excluded from the structure, is determined as:

$$S_{k-1}(a_{ij}^{k}) = g(\mathcal{E}_k) - g(\mathcal{E}_k \setminus a_{ij}^{k}).$$

The *significance* $S_{k+1}(a_{ij}^{k})$ *of the arc* $a_{ij}^{k} \in \mathcal{A} \setminus \mathcal{E}_k$ between nodes X_i and X_j, that is, the arc candidate to be included in the structure, is determined as:

$$S_{k+1}(a_{ij}^{k}) = g(\mathcal{E}_k \cup a_{ij}^{k}) - g(\mathcal{E}_k).$$

Figure 5.3 and Figure 5.4 show the proposed sequential algorithms in order to learn Bayesian network structures. The adapted SFFS and SBFS try to

Adapted SBFS

Step 1: Exclusion

Using the basic greedy method, remove arc a_{ij}^{k+1} from set $\mathcal{E}_k$, i.e. delete the least significant arc a_{ij}^{k+1} from set $\mathcal{E}_k$ to form a graph $\mathcal{E}_{k+1} = \mathcal{E}_k \setminus a_{ij}^{k+1}$. If it is not possible to exclude an arc, stop.

Step 2: Conditional inclusion

Find the most significant arc in the set of excluded arcs $A \setminus \mathcal{E}_{k+1}$. If a_{ij}^{k+1} is the most significant arc in set $\mathcal{E}_{k+1}$, i.e. $g(\mathcal{E}_{k+1} \cup a_{ij}^{k+1}) \geq g(\mathcal{E}_{k+1} \cup a_{ij}^{l})$ $\forall l = 1, 2, \ldots, k$, then set $k = k + 1$ and return to *Step 1*, but if a_{ij}^{r} $1 \leq r \leq k$ is the most significant arc in set $\mathcal{E}_{k+1}$, i.e. $g(\mathcal{E}_{k+1} \cup -a_{ij}^{r}) > g(\mathcal{E}_k)$, then include a_{ij}^{r} to $\mathcal{E}_{k+1}$ to form a new graph $\mathcal{E}'_k$, i.e. $\mathcal{E}'_k = \mathcal{E}_{k+1} \cup a_{ij}^{r}$. Note that now $g(\mathcal{E}'_k) > g(\mathcal{E}_k)$. If $k = 2$, then set $\mathcal{E}_k = \mathcal{E}'_k$ and $g(\mathcal{E}_k) = g(\mathcal{E}'_k)$ and return to *Step 1*. If not, go to *Step 3*.

Step 3: Continuation of conditional inclusion

Find the most significant arc a_{ij}^{s} in the set of excluded arcs $A \setminus \mathcal{E}'_k$. If $g(\mathcal{E}'_k \cup a_{ij}^{s}) \leq g(\mathcal{E}_{k-1})$ then set $\mathcal{E}_k = \mathcal{E}'_k$, $g(\mathcal{E}_k) = g(\mathcal{E}'_k)$ and return to *Step 1*. If $g(\mathcal{E}'_k \cup a_{ij}^{s}) > g(\mathcal{E}_{k-1})$, then include a_{ij}^{s} to set $\mathcal{E}'_k$ to form a new graph $\mathcal{E}'_{k-1}$, i.e. $\mathcal{E}'_{k-1} = \mathcal{E}'_k \cup a_{ij}^{s}$. Set $k = k - 1$. Now if $k = 2$, then set $\mathcal{E}_k = \mathcal{E}'_k$ and $g(\mathcal{E}_k) = g(\mathcal{E}'_k)$ and return to *Step 1*. If not, repeat *Step 3*.

Fig. 5.4. Adapted sequential backward floating selection algorithm.

avoid the main problem of all the greedy methods, including the classic B-algorithm: the nesting problem.

The adapted floating search algorithms follow the general scheme of the original methods. For instance, the forward algorithm starts with the *inclusion* step, where it is proposed that the most significant arc be part of the network structure $\mathcal{E}_k$ to form $\mathcal{E}_{k+1}$. At the *conditional exclusion* step, it is proposed that the least significant arc of $\mathcal{E}_{k+1}$ be excluded. If it is the last arc introduced, exclusion is rejected. If not, the arc is deleted to form $\mathcal{E}'_k$ and the *continuation of the conditional exclusion* is carried out. At this stage, the algorithm asks for the least significant arc of the network structure $\mathcal{E}'_k$. If the value of the scoring function of the structure without the least significant arc is lower than the last time the structure had the same number of arcs, the exclusion is rejected and the method returns to the inclusion step. If not, the arc is deleted to form $\mathcal{E}'_{k-1}$ and the deletion of another arc is proposed.

In contrast to other sequential algorithms, by means of these methods, the number of added arcs does not need to be previously fixed; it 'floats' during the search process.

It must be noted that a Bayesian network structure is represented by means of a directed acyclic graph. When adding an arc, if the resulting graph is cyclic, the arc is rejected and another arc is selected. For the sake of simplicity, this

rejection of an arc when cyclicity appears is not written in Figure 5.3 and Figure 5.4.

In the original SFFS and SBFS, the improvement of the total score, neither in addition nor in deletion, is considered. It is possible that, by adding or deleting a feature, the score metric gets worse. In adapted algorithms, a threshold of this worsening is fixed. This threshold is used as follows: when an arc is added (or removed), if the current score is worse than the fixed threshold, the arc is rejected and another arc is proposed to carry out the movement. However, it is possible that a small deterioration in the score measure could guide the search process to a different area of the search space and, thus, to new solutions improving the scoring function. The threshold decreases with the number of visited solution candidates. For simplicity, the use of the score is not presented in Figure 5.3 and Figure 5.4. It must be noted that when the threshold is fixed at 0, the adapted algorithms become classic greedy methods.

5.2.3 Experimental results

In order to compare the behaviour of both floating methods, they are tested using the K2 scoring function Cooper and Herskovits, 1992 over four Bayesian network databases from the literature:

- The *Alarm* Beinlinch et al., 1989 network is considered a benchmark to evaluate Bayesian network learning algorithms. It has 37 nodes, 46 arcs and it is related to medical diagnosis.
- The *Hailfinder* Abramson et al., 1996 network has 56 variables, 66 arcs and it was designed to forecast severe weather in North-Eastern Colorado, USA.
- The *Insurance* Binder et al., 1997 network has 27 nodes and 52 arcs and it evaluates the risk in car insurance.
- The *Mildew* Jensen and Jensen, 1996 network, composed of 35 nodes and 46 arcs, decides how much fungicide should be used against a mildew attack on wheat.

It must be noted that, whereas the experimentation with *Alarm* is performed over the first 3000 cases of the well-known database proposed by Cooper and Herskovits (1992), the experiments with *Hailfinder*, *Insurance* and *Mildew* networks are carried out over 10000 cases simulated from original Bayesian networks.

In order to compare the results of the adapted floating methods with the results of a 'standard' and 'benchmark' sequential algorithm, a comparison with the B algorithm is performed. Moreover, the B3 algorithm is proposed in order to make an extensive comparison. The B and B3 algorithms are explained in Section 4.3. The adapted SFFS is compared with the *forward* B and B3 algorithms which, starting with an empty solution, iteratively construct a Bayesian network structure by means of the operators. The adapted SBFS is compared with the *backward* B and B3 algorithms. In this case, the search

		Forward			*Backward*		
		B	*B3*	*SFFS*	*B*	*B3*	*SBFS*
Empty	*score*	-14520.2	-14440.2	-14472.9			
	n.eval	61832	69228	255999			
Full	*score*				$-20259.2 \pm 496.1^{\dagger}$	-16288.51 ± 299.1	-18149.7 ± 488.7
	n.eval				66359.3 ± 2341.5	$247788.5 \pm 8761.7^{\dagger}$	$157540.0 \pm 3108.9^{\dagger}$
Random	*score*			-14790.2 ± 225.8			-14885.7 ± 396
	n.eval			203378.3 ± 9203.1			205335.7 ± 9391.1
B-noise	*score*			-14473.0 ± 0.2			$-15256.2 \pm 389.5^{\dagger}$
	n.eval			$244762.8 \pm 912.5^{\dagger}$			187655.0 ± 9929.7

Table 5.1. Score results and number of visited solutions of the networks induced by the search algorithms over 10 runs (when feasible) in a simulation of the *Alarm* network. The real value of the K2 scoring function is -14412.69.

process starts with a proposed network solution: at each step the B algorithm removes the arc with the major score deterioration from the structure.

In order to perform the comparison of the backward methods, ten complete structures are generated. To create each of these structures, a randomly generated ancestral ordering is used.

Moreover, due to the fact that a partial solution may be required to start the original floating algorithms, the experimentation is enlarged for the adapted floating methods, starting the search process in random structure of the search space between the empty and the full structures. These structures are generated by means of two different processes. First, the networks are randomly generated as follows: starting with an empty solution structure, an arc is added to the network structure with a fixed probability. This initialisation is called *Random* initialisation, and the probability of the appearance of an arc is fixed at 0.05. The random generation of structures may produce non-valid Bayesian networks. To overcome this difficulty, a simple 'repairing' operator is used: when a cycle is detected in the structure, one arc from the cycle is deleted randomly. Ten independent structures are generated as the initialisation network of floating methods. The improvement threshold is fixed at $0.10n$, where n is the number of variables of the domain.

The second approach to generate the initial structures is based on the network induced by the forward B algorithm called *B-noise* initialisation. When a forward floating algorithm is used, if an arc is selected by the B algorithm, it could be deleted with a certain probability. When a backward floating algorithm is used, if an arc is not selected by the B algorithm, it could be added with a certain probability. The idea is to produce networks with fewer arcs when the forward methods are carried out, and with more arcs when the backward methods are carried out. In the case of the forward method, the probability of deletion is fixed at 0.10. In the case of the backward method, the probability of addition is fixed, assuming that the expected number of deleted arcs for the forward algorithm is the same as the expected number of added arcs for the backward engine.

Tables 5.1, 5.2, 5.3 and 5.4 show the results in terms of score and number of evaluated solutions during the search process. When more than one run

		Forward		
		B	*B3*	*SFFS*
Empty	*score*	−217105.1	−217072.6	−217085.64
	n.eval	208227	214974	304616

Table 5.2. Score results and number of visited solutions of the networks induced by the search algorithms over 10 runs (when feasible) in a simulation of the *Hailfinder* network. The real value of the K2 scoring function is −217663.39.

		Forward			*Backward*		
		B	*B3*	*SFFS*	*B*	*B3*	*SBFS*
Empty	*score*	−58964.3	−58641.6	−58964.3			
	n.eval	28435	36000	75294			
Full	*score*				−67162.1 ± 2111.63†	−59783.3 ± 159.9	−64881.0 ± 762.1†
	n.eval				22032.5 ± 182.4	67009.8 ± 4052.7†	52255.8 ± 1341.2†
Random	*score*			−60068.6 ± 669.0			−60585.5 ± 383.4†
	n.eval			69245.0 ± 2404.2			69488.1 ± 4112.9
B-noise	*score*			−59932.4 ± 241.8			−60502.1 ± 231.6
	n.eval			61924.2 ± 2815.8†			54594.9 ± 4917.4

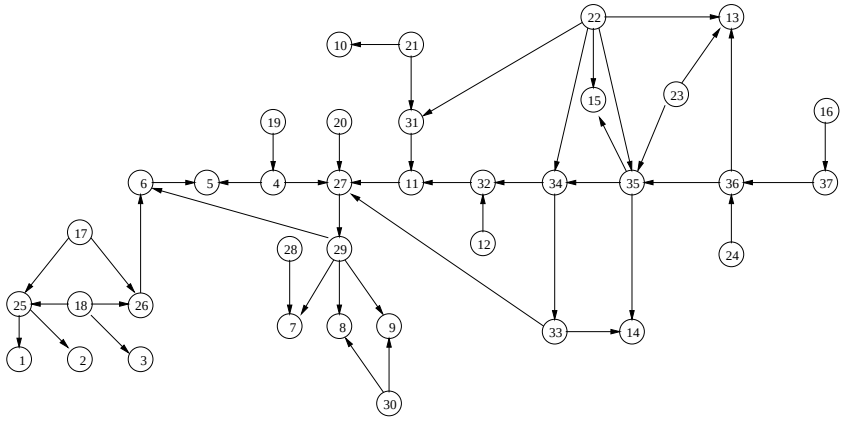
Table 5.3. Score results and number of visited solutions of the networks induced by the search algorithms over 10 runs (when feasible) in a simulation of the *Insurance* network. The real value of the K2 scoring function is −59257.60.

		Forward			*Backward*		
		B	*B3*	*SFFS*	*B*	*B3*	*SBFS*
Empty	*score*	−202685	−202199	−202655			
	n.eval	49650	61208	210159			
Full	*score*				−239320 ± 5242†	−205443 ± 2268	−212818 ± 2280†
	n.eval				12266 ± 234	90096 ± 2009†	155897 ± 4289†
Random	*score*			−203862 ± 1363			−203584 ± 1082
	n.eval			202755 ± 9537			201431 ± 13504
B-noise	*score*			−203639 ± 2726†			−202660 ± 102
	n.eval			180420 ± 8625			213539 ± 9731†

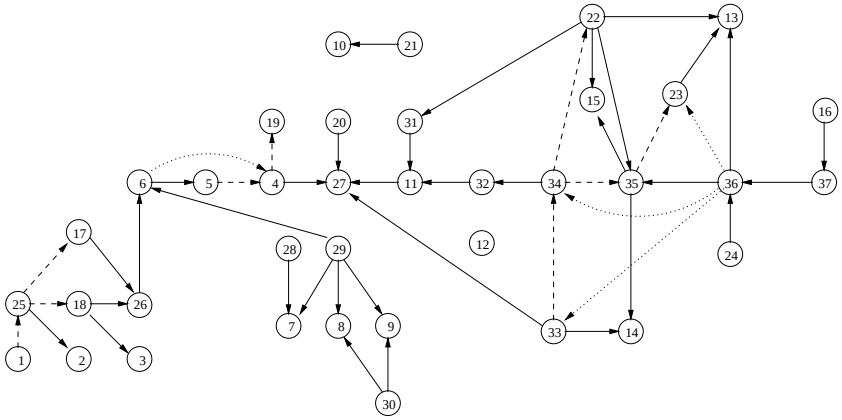
Table 5.4. Score results and number of visited solutions of the networks induced by the search algorithms over 10 runs (when feasible) in a simulation of the *Mildew* network. The real value of the K2 scoring function is −205668.23.

is performed, the average score (and its standard deviation) and the average number of evaluated solutions (and its standard deviation) are displayed. The empty blocks on the tables are unfeasible combinations of initial structures and search engines. It must be noted that for the *Hailfinder* domain, the problem is computationally treatable only when the search engines start with an empty solution.

In order to carry out a more thorough analysis of the results, the statistical significance of the obtained differences is studied when several runs are performed. The Mann-Whitney Mann and Whitney, 1947 test is carried out to determine the significance of the differences in Tables 5.1, 5.2, 5.3 and 5.4. For each initialisation, statistically significant differences with respect to the best results attained are marked. The symbol † denotes statistically significant differences at the 0.05 confidence level in relation to the best results reached by an algorithm in a certain initialisation scheme.



(a) *Alarm* structure



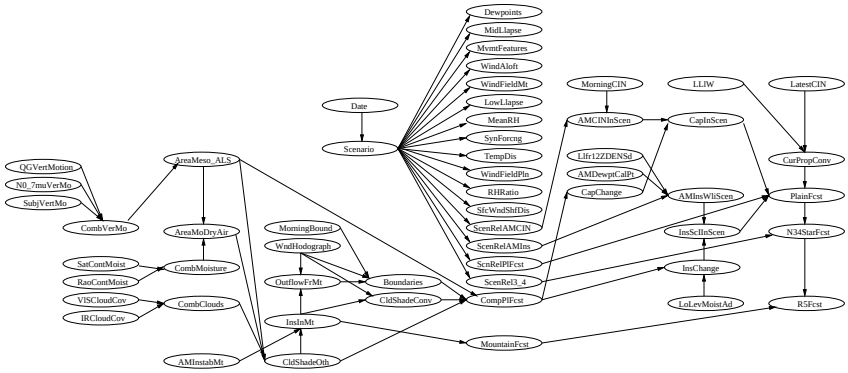
(b) Network structure learned by SFFS

Fig. 5.5. The original *Alarm* network structure has 37 nodes and 46 arcs. The network structure learned by SFFS, starting with an empty solution, has 47 arcs: 4 extra arcs (dotted line), 8 reverse arcs (dashed line) and 3 missing arcs.

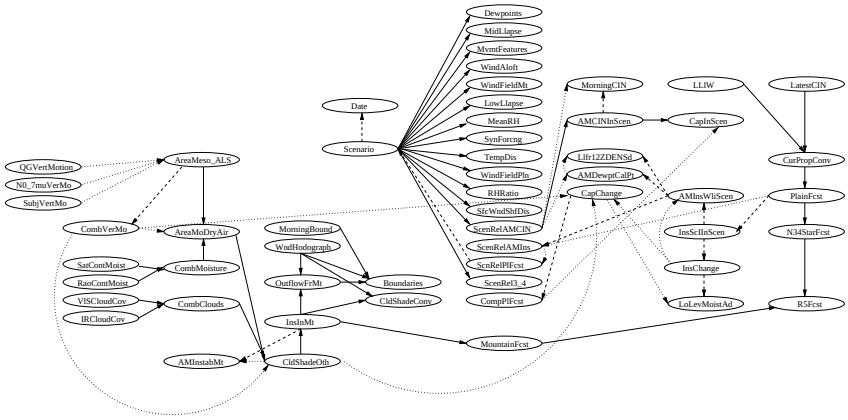
The comparison is made in three steps: (i) forward methods with an 'empty' initialisation, (ii) backward methods with a 'full' initialisation and (iii) floating methods with 'middle' initialisation.

For the forward methods starting with an empty initialisation, the *Alarm*, *Hailfinder*, *Insurance* and *Mildew* domains show a similar behaviour. The floating method, requiring a higher number of evaluated solutions, improves the scoring function values of the B algorithm without statistically significant differences. However, the B3 algorithm proposed attains slightly better fitness results than the floating algorithm.

The results obtained by backward methods, starting with a full initialisation, should be studied for each problem domain separately.



(a) *Hailfinder* structure



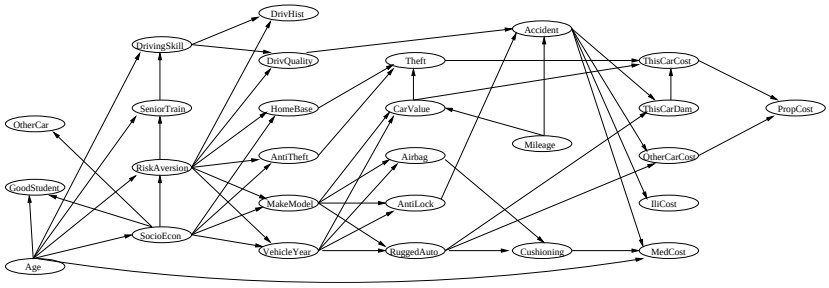
(b) Network structure learned by SFFS

Fig. 5.6. The original *Hailfinder* network structure has 56 nodes and 66 arcs. The network structure learned by SFFS, starting with an empty solution, has 69 arcs: 17 extra arcs (dotted line), 13 reverse arcs (dashed line) and 14 missing arcs.

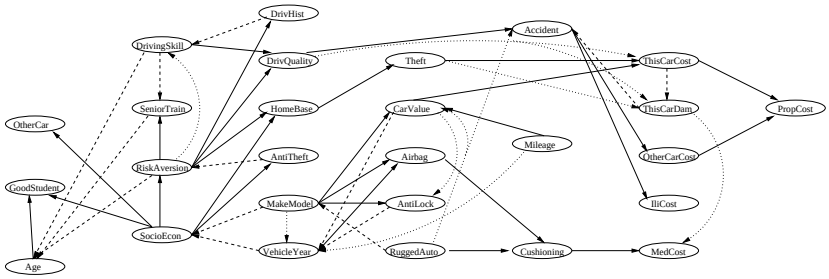
In the *Alarm* domain, although the B algorithm requires the lowest number of evaluated structures with statistically significant differences, it attains the lowest score value. The B3 algorithm achieves the best scoring results without significant difference with respect to the floating method.

In the *Insurance* and *Mildew* domains, the B algorithm needs the lowest number of evaluated solutions (with statistically significant differences) to reach the lowest values of the score function. The best value of the scoring measure is obtained by the B3 algorithm with statistically significant differences with respect to floating methods and the B algorithm.

When the floating methods are compared, forward floating seems to attain slightly better results than the backward methods. For example, in the *Alarm* domain, when B-noise initialisation is used, the forward method requires a



(a) *Insurance* structure

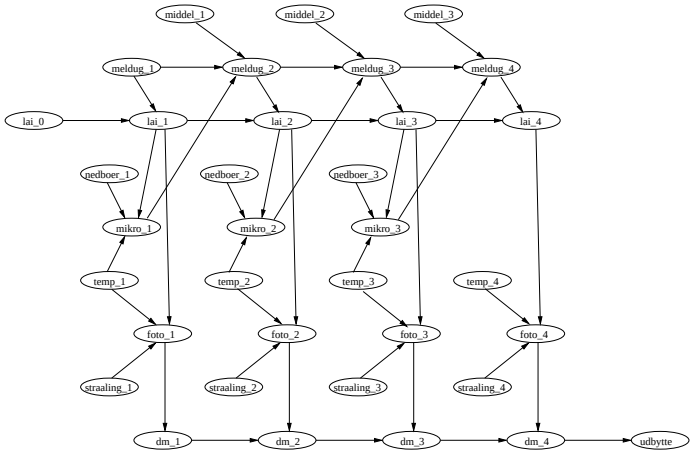


(b) Network structure learned by SFFS

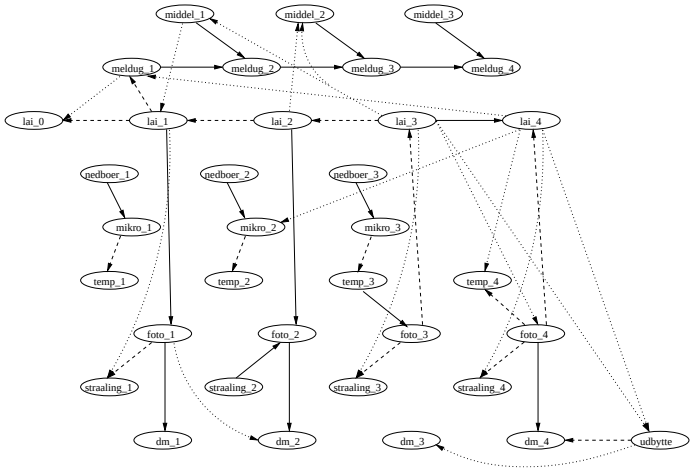
Fig. 5.7. The original *Insurance* network structure has 27 nodes and 52 arcs. The network structure learned by SFFS, starting with an empty solution, has 49 arcs: 13 extra arcs (dotted line), 11 reverse arcs (dashed line) and 14 missing arcs.

statistically significant higher number of evaluated structures to reach a statistically significant best scoring function. However, when Random initialisation is proposed, statistically significant differences are not found for scoring values and the required number of evaluated solutions. A similar behaviour could be observed in the *Insurance* domain. Although the forward method needs a statistically significant higher number of evaluated networks for B-noise initialisation, it attains better but not statistically significant scoring results with respect to the backward algorithm. Finally, in the *Mildew* problem domain, for B-noise initialisation, a statistically significant worse value of the scoring measure is attained by the forward method with a statistically significant lower number of evaluated solutions. For Random initialisation, the score results and the number of evaluated structures are so close that no statistically significant difference could be found.

In order to perform a qualitative study of the solutions learned by the *forward* floating method, Figures 5.5, 5.6, 5.7 and 5.8 respectively show the original network structures and the network structures induced by the forward floating algorithm for the *Alarm*, *Hailfinder*, *Insurance* and *Mildew* networks. The differences between the original networks and the learned networks are stronger when the number of nodes and the number of arcs grow. Conse-



(a) *Mildew* structure



(b) Network structure learned by SFFS

Fig. 5.8. The original *Mildew* network structure has 35 nodes and 46 arcs. The network structure learned by SFFS, starting with an empty solution, has 46 arcs: 16 extra arcs (dotted line), 14 reverse arcs (dashed line) and 16 missing arcs.

quently, the *Alarm* structure learned by floating is the one closest to the original with, 4 extra arcs, 8 reverse arcs and 3 missing arcs.

In general, floating methods obtain competitive results when compared with benchmark methods, the B and B3 algorithms. The promising results in the field of feature subset selection are not so favourable in learning Bayesian network structures. Although well-known techniques like the B3 algorithm overcome the score value of floating methods, this experimentation suggests that other decisions could be taken to improve the performance of floating

```
GRASP
   InitialiseSolution();
   while stopping criterion is not met
      Solution = ConstructGreedyRandomised(Solution);
      Solution = LocalSearch(Solution);
      Update(Solution, BestSolution);
   end while;
   return BestSolution;
```

Fig. 5.9. General scheme of the GRASP algorithm.

methods. Other scoring functions or another search space could explore several solutions and take advantage of the intrinsic characteristics of floating methods.

5.3 GRASP to learn Bayesian network structures

The Greedy Randomized Adaptive Search Procedure (GRASP) Feo and Resende, 1989; Feo and Resende, 1995 is proposed in the classic optimisation field. It combines a local search with a solution construction in an iterative process. In this way, the GRASP method tries to avoid the local optimum solutions.

The GRASP optimisation method is successfully used in several problems such as set covering problems arising from the incidence matrix of Steiner triple systems Feo and Resende, 1989, maximum independent set problem Feo et al., 1989, corporate acquisition of flexible manufacturing equipment Bard and Feo, 1991, computer aided process planning Bard and Feo, 1989, airline flight scheduling and maintenance base planning Feo and Bard, 1989, scheduling of parallel machines Laguna and González-Velarde, 1991, set packing problems Delorme et al., 2004 and matrix bandwidth maximisation Piñana et al., 2004. Although GRASP is a method widely used in others fields, as far as we know it has been never applied to Bayesian networks learning.

5.3.1 The original Greedy Randomized Adaptive Search Procedure

The GRASP method is an iterative sequential search method with two steps: a *construction* step and a *local improvement* step. The best solution found in the entire search is returned as the final result. The basic algorithm of the GRASP method is given in Figure 5.9.

At the construction step, a feasible solution is iteratively constructed in a semi-greedy way. A set of possible element candidates to be part of the

```
Construction
    InitialiseSolution();
    while Construction not finished
        RCL = ChooseBestCandidates(Solution);
        s = SelectElementRandomly(RCL);
        Solution = Solution ∪ s;
        AdaptGreedyFunction(s);
    end while;
    return Solution;
```

Fig. 5.10. Construction step of the GRASP method.

solution is recalculated at each iteration. These candidates are a 'piece' of the induced solution. One element selected from this set of candidates is added to the solution. The element candidate list is evaluated with respect to a scoring function. The set is usually formed by the r 'best' elements. The method is called *adaptive* as a result of the modification of the improvements associated with every element selected to be part of the solution. The aim of this adaptation is to reflect the changes produced by the addition of the element to the solution.

The *randomised* choice of one element of the candidate list to be part of the solution provides the random component of the algorithm. The set of r best candidates is called the *restricted candidate list* (RCL). Figure 5.10 shows the general algorithm of the construction step.

The solution induced in the construction step is not guaranteed to be a local optimum with respect to a simple neighbourhood definition. Therefore, the use of a classical local sequential search could significantly benefit the final result of the search process. A local search algorithm looks within a restricted search space (neighbourhood of the current solution), replacing the current solution with the best one found in the restricted space. A suitable election of the neighbourhood space, the starting solution and the search engine is the crucial point of any local sequential search procedure. Given a neighbourhood space $\mathcal{N}$, a local search could be seen as Figure 5.11 displays.

This classic local sequential search scheme could be generalised to a large number of domains changing the neighbourhood $\mathcal{N}$. Taking the representation of the solution into account, several neighbourhoods could be proposed.

One appealing characteristic of the GRASP procedure is the few number of parameters required: only the size of the RCL and the stopping criterion.

The general scheme of GRASP could be specialised and enlarged with other optimisation techniques, like path relinking Laguna and Martí, 1999 and long-term memory Fleurent and Glover, 1999. However, the basic GRASP proposed is developed in this work.

```
Local search
   while not local optimum
      Neighbour = FindBetterSolution(𝒩(Solution));
      Solution = neighbour;
   end while;
   return Solution;
```

Fig. 5.11. Local search for the GRASP method.

5.3.2 The Greedy Randomized Adaptive Search Procedure to learn Bayesian networks

In order to learn Bayesian networks by means of the GRASP method, the general scheme presented in Figure 5.9 is followed. The adaptation to learning Bayesian networks is performed in both the construction step and the local improvement step.

During the search of the Bayesian network structure, the construction step is similar to the B algorithm, but the arc to be added is randomly selected from the RCL. The list of candidates to be added as a part of the constructed solution is composed of the r arcs which obtain the best score when they are added to the previous directed acyclic graph. This scheme is repeated until no improvement of the score metric is achieved. In order to reach a better fitting Bayesian network structure, the initial solution of the construction step is the structure induced at the previous local search step.

At the local improvement step, the selection of the neighbourhood is a crucial task. In this work, bearing in mind the connectivity matrix used to represent a solution, one of the most obvious neighbourhoods is composed of the feasible structures whose connectivity matrix is located at Hamming distance 1. Nevertheless, a better neighbourhood takes into account three possible operations over the current connectivity matrix: arc addition, arc deletion and arc reversal. This implies that the neighbourhood $\mathcal{N}$ is composed of all the structures that are different in an extra arc, a missing arc or a reverse arc. Thus, the local improvement step is performed by means of the B3 algorithm.

5.3.3 Experimental results

In order to compare the behaviour of the GRASP method, it is tested with the K2 scoring function Cooper and Herskovits, 1992 over four Bayesian network databases from the literature: *Alarm* Beinlinch et al., 1989, *Hailfinder* Abramson et al., 1996, *Insurance* Binder et al., 1997 and *Mildew* Jensen and Jensen, 1996. The databases are used in the same way as in Section 5.2.3.

		B	B3	GRASP		
				minimum	maximum	average±sd
Alarm	score	−14520.25	−14440.23	−14500.56	−14361.94	−14424.75±38.68
	number eval.	61832	69228	83886	95323	92409.80±6837.14
Hailfinder	score	−217105.08	−217072.60	−217416.67	−217122.75	−217279.72±87.65
	number eval.	208227	214974	218246	239054	250326.01±19789.66
Insurance	score	−58964.35	−58641.59	−59142.77	−58492.28	−58837.23±262.84
	number eval.	28435	36000	40008	45998	42249.80±3371.92
Mildew	score	−202685.68	−202199.62	−202602.34	−201563.43	−202125.52±288.47
	number eval.	49650	61208	73225	76961	73723.60±3069.62

Table 5.5. Results of the scores and the number of evaluated solutions required for the stopping criterion of the *Alarm*, *Insurance*, *Hailfinder* and *Mildew* networks.

With the purpose of comparing them with the 'standard' sequential benchmark algorithms, the B and B3 algorithms are run. These algorithms are explained in Section 4.3.

Whereas the B and B3 algorithms do not require any parameters, GRASP needs to fix the number of RCL candidates. Although an extensive experimentation is performed to choose an appropriate RCL size (in our experiments the size of RCL varies from 2 to 10), the results are only shown when r is fixed at 5. Similar results are achieved with other values of r. Ten independent runs are executed for each problem.

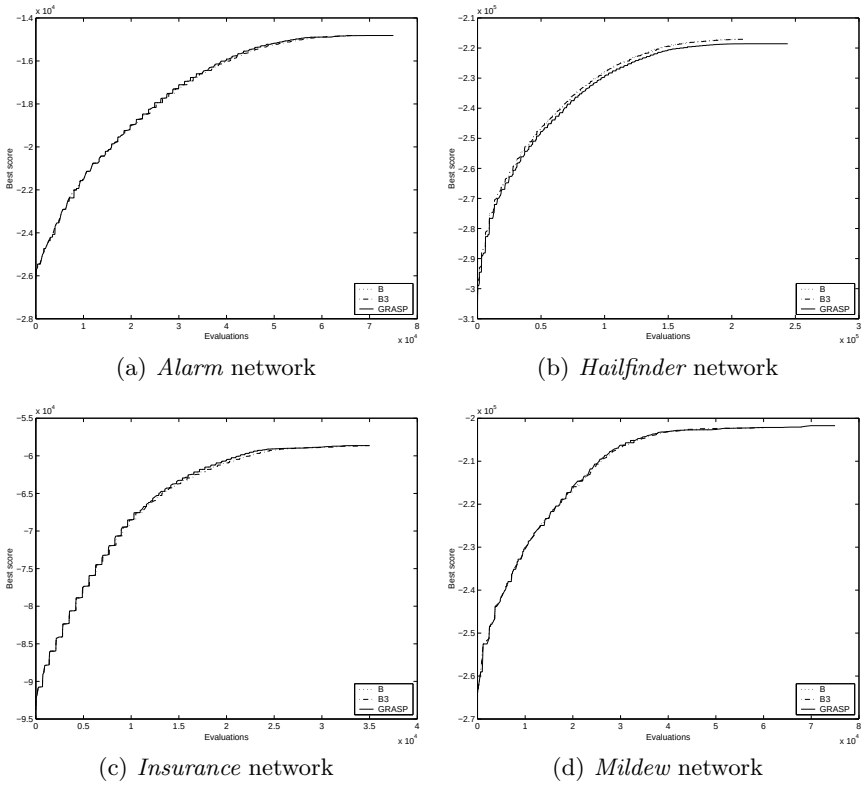
Table 5.5 displays the obtained efficacy and efficiency results for the *Alarm*, *Hailfinder*, *Insurance* and *Mildew* domains. For each search engine, the reached scoring measure and the required number of solutions evaluated are shown. In the case of GRASP, as ten runs are executed, the mean and the standard deviation are shown. Table 5.5 also shows the minimum and maximum scoring measures achieved by GRASP and their corresponding number of evaluated solutions over the ten executed independent runs. The real value of the K2 scoring function for the *Alarm*, *Hailfinder*, *Insurance* and *Mildew* databases is −14412.69, −217663.39, −59257.60 and −205668.23 respectively.

In order to carry out a deeper analysis of the results, the statistical significance of the differences obtained is studied. The binomial sign test is carried out to determine the significance of the score differences among the single values obtained by the B and B3 algorithms and the multiple values achieved by GRASP for each domain problem.

The results obtained by GRASP should be studied separately for each dataset.

The *Alarm* and *Mildew* domains show similar behaviour for the three proposed algorithms. Although the B algorithm needs the lowest number of evaluated structures, it attains the lowest score with a statistically significant difference whose p-value is $p = 0.002$. The B3 algorithm also needs a lower number of evaluated solutions in comparison with those needed by GRASP. Although the results of B3 and GRASP do not show statistically significant differences, the scoring value of B3 is worse than the average results obtained by GRASP.

In the *Hailfinder* domain, the GRASP search engine behaves surprisingly. With the lowest number of evaluated solutions, the B and B3 algorithms



(a) *Alarm* network (b) *Hailfinder* network

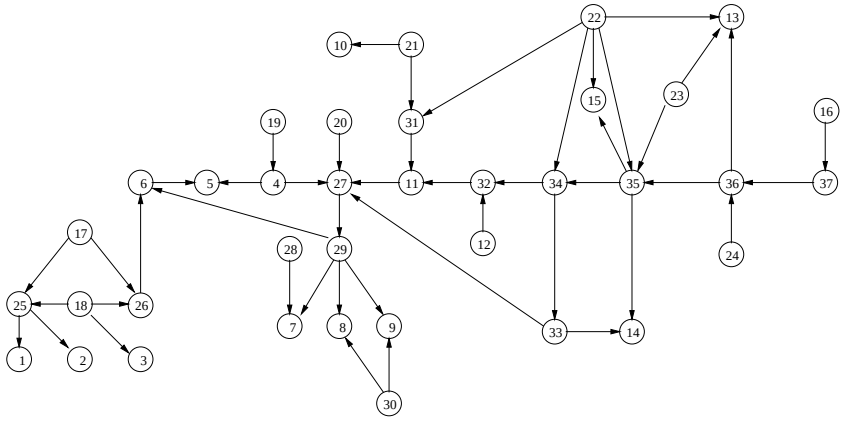
(c) *Insurance* network (d) *Mildew* network

Fig. 5.12. Evolution of the best scoring function in B3 and a GRASP typical run of (a) the *Alarm*, (b) *Hailfinder*, (c) *Insurance*, and (d) *Mildew* networks.

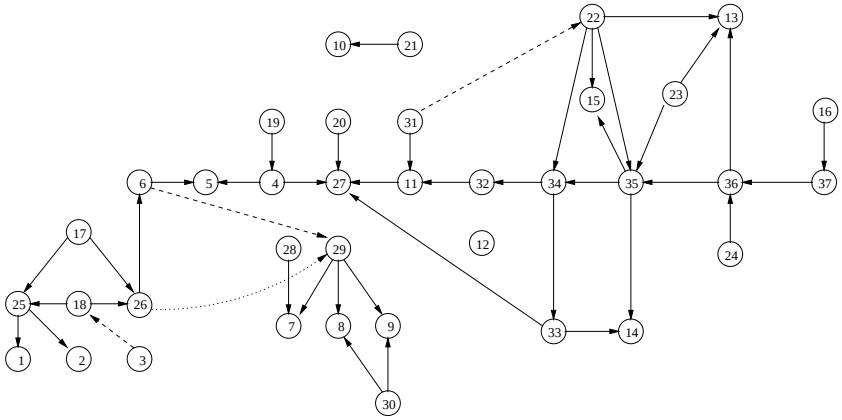
achieve higher score values than GRASP, with statistically significant differences whose p-value is $p = 0.002$. This result seems to be related to the size of the r parameter of the GRASP search method. In this domain it has been stated that GRASP obtains better results in comparison with those reached by the B and B3 algorithms, when $r < 5$.

In the *Insurance* domain, the B and B3 algorithms also require a lower number of evaluated structures in comparison with those needed by GRASP. Although the mean value of the scoring function achieved by GRASP is higher than the B score, statistically significant differences are not found. GRASP attains better but not statistically significant scoring results with respect to B3.

Figure 5.12 shows, for the four databases, the evolution of the best scores found in the search process with respect to the number of evaluated solutions in a typical GRASP run: (a) *Alarm*, (b) *Hailfinder*, (c) *Insurance* and (d) *Mildew*. It can be observed that the induced scoring curves of the four data-



(a) *Alarm* structure

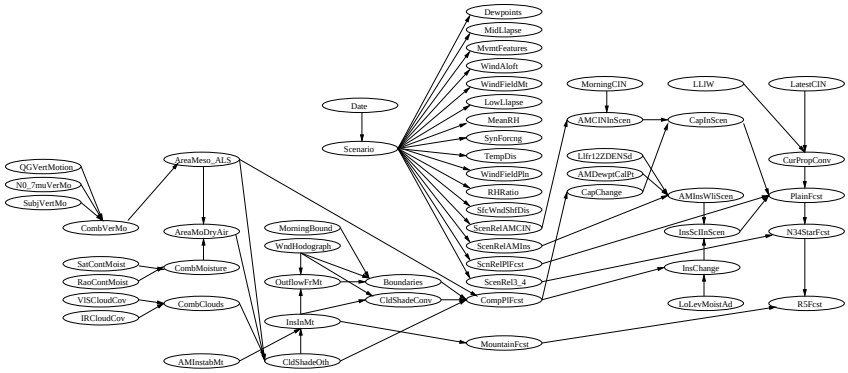


(b) Network structure learned by GRASP

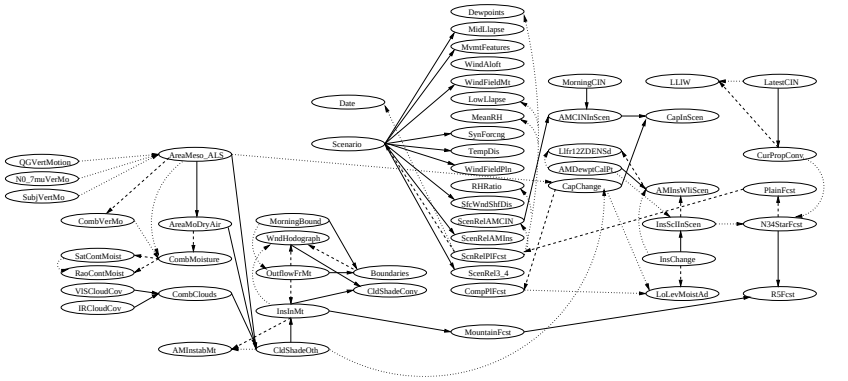
Fig. 5.13. The original *Alarm* network structure has 37 nodes and 46 arcs. The network structure learned by GRASP has 44 arcs: 1 extra arc (dotted line), 3 reverse arcs (dashed line) and 3 missing arcs.

bases increase logarithmically. In this way, the *Alarm*, *Insurance* and *Mildew* domains take advantage of the extra power of the GRASP engine. A poor behaviour is shown by GRASP in the *Hailfinder* dataset.

In order to carry out a qualitative study of the solutions learned by GRASP, Figures 5.13, 5.14, 5.15 and 5.16 display the original network structures and the closer network learned by a GRASP run. It can be observed that a major complexity of the problem (in number of nodes and number of arcs) produces a major differences between the real and the learned structure. A comparison between the learned structures and the real networks is performed. The *Alarm* structure learned by GRASP is the one closest to the real network, with 1 extra arc, 3 reverse arcs and 3 missing arcs. Consistently,



(a) *Hailfinder* structure

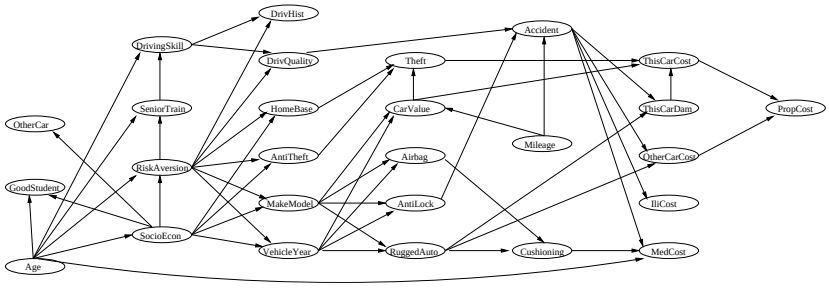


(b) Network structure learned by GRASP

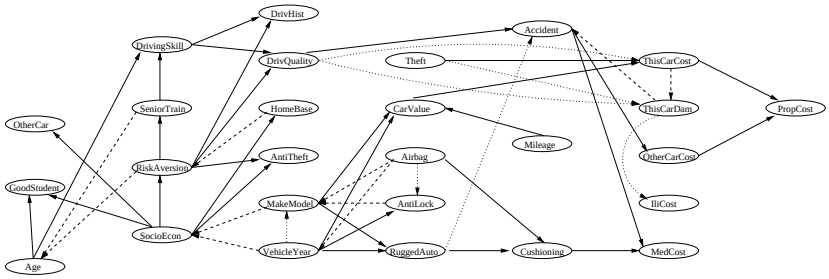
Fig. 5.14. The original *Hailfinder* network structure has 56 nodes and 66 arcs. The network structure learned by GRASP has 70 arcs: 25 extra arcs (dotted line), 16 reverse arcs (dashed line) and 21 missing arcs.

the structure induced to the *Hailfinder* domain is the one most distant from the real network, with 25 extra arcs, 16 reverse arcs and 21 missing arcs. The *Insurance* and *Mildew* learned networks, with 8 and 13 extra arcs, 10 and 15 reserve arcs and 13 and 12 missing arcs, are not so distant from the original network structures.

In order to summarise the obtained results, it must be noted that GRASP is not as competitive as expected. In spite of the extra computational cost, the maximum score values attained by GRASP are slightly better than the values obtained by the B and B3 algorithms. Nevertheless, the average score values are similar to the values of B and B3. However, improvements of the scheme followed in this work could be proposed to enhance the results. The size of the RCL, the neighbourhood of the local search and the stopping criterion are good candidates for these improvements.



(a) *Insurance* structure



(b) Network structure learned by GRASP

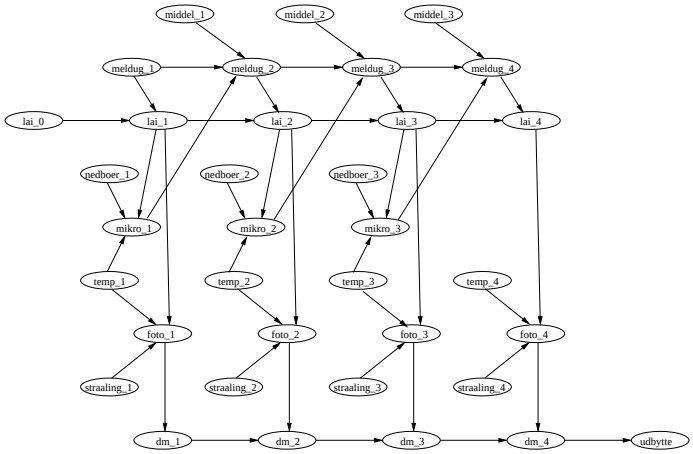
Fig. 5.15. The original *Insurance* network structure has 27 nodes and 52 arcs. The network structure learned by GRASP has 47 arcs: 8 extra arcs (dotted line), 10 reverse arcs (dashed line) and 13 missing arcs.

5.4 Estimation of distribution algorithms to learn Bayesian network structures

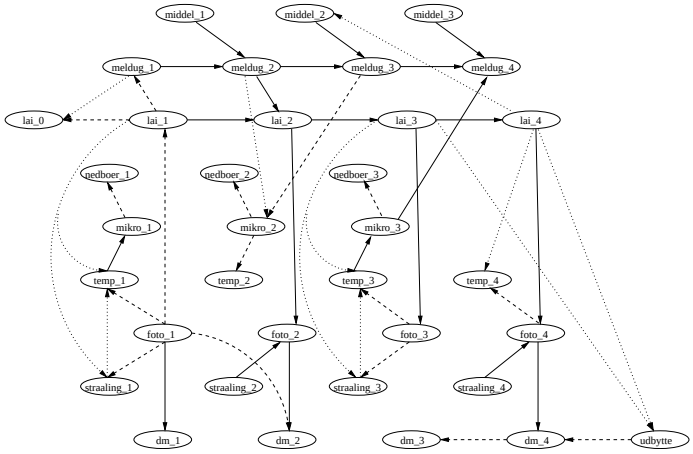
The estimation of distribution algorithms (EDAs) Larrañaga and Lozano, 2001; Mühlenbein and Paaß, 1996 are a population-based, stochastic search method. Instead of crossover and mutation operators, they estimate the joint distributions of the promising solutions. This estimate is used to generate new individuals –see Chapter 3 for a deeper introduction to EDAs–. Depending on the objective function, any optimisation task could be carried out. The main problem with EDAs is how the probability distribution $p_l(\boldsymbol{x})$ is estimated. Obviously, the computation of all the parameters needed to specify the probability distribution is impractical. This has led to several approximations where the probability distribution is assumed to factorise according to a probability model.

5.4.1 The Univariate Marginal Distribution Algorithm and Population Based Incremental Learning algorithm

The Univariate Marginal Distribution Algorithm (UMDA), introduced by Mühlenbein (1998), is a simple paradigm to estimate the joint probability



(a) *Mildew* structure



(b) Network structure learned by GRASP

Fig. 5.16. The original *Mildew* network structure has 35 nodes and 46 arcs. The network structure learned by GRASP has 47 arcs: 13 extra arcs (dotted line), 15 reverse arcs (dashed line) and 12 missing arcs.

distribution of the selected individuals at each generation. It is factorised as a product of independent univariate marginal distributions:

$$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) = \prod_{i=1}^{n} p_l(x_i)$$

Figure 5.17 shows the pseudocode for the UMDA approach. Santana and Ochoa (1999) and Santana et al. (2000) propose modifications of the basic UMDA at the simulation step. Mahnig and Mühlenbein (2000) and

EDA
$D_0 \leftarrow$ Generate M individuals (the initial population) randomly
repeat for $l = 1, 2, \ldots$ **until** a stop criterion is met
$D^s_{l-1} \leftarrow$ Select $N \leq M$ individuals from D_{l-1} according to a selection method
$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D^s_{l-1}) = \prod_{i=1}^{n} p_l(x_i) = \prod_{i=1}^{n} \frac{\sum_{j=1}^{N} \delta_j(X_i = x_i | D_l^{Se} - 1)}{N} \leftarrow$ Estimate the joint probability of selected individuals
$D_l \leftarrow$ Sample M individuals (the new population) from $p_l(\boldsymbol{x})$

Fig. 5.17. Pseudocode for the UMDA algorithm.

Mühlenbein and Mahnig (2000) perform a mathematical analysis of this EDA model.

Population Based Incremental Learning (PBIL) Baluja, 1994 is another paradigm which carries out a population-based, stochastic search. The objective is to obtain the optimum of a function defined in the binary space $\Omega = \{0, 1\}^n$ (the next explanations could easily be extended to non-binary search spaces). At each generation, the population of individuals is represented by a vector of probabilities: $p_l(\boldsymbol{x}) = (p_l(x_1), \ldots, p_l(x_i), \ldots, p_l(x_n))$ where $p_l(x_i)$ refers to the probability of obtaining a value of 1 in the i^{th} component of D_l, the population of individuals in the l^{th} generation. The algorithm works as follows (see Figure 5.18). At each generation, using the probability vector, $p_l(\boldsymbol{x})$, M individuals are obtained. Each of these M individuals are evaluated and the N best of them ($N \leq M$) are selected. They are denoted by $\boldsymbol{x}^l_{1:M}, \ldots, \boldsymbol{x}^l_{i:M}, \ldots, \boldsymbol{x}^l_{N:M}$. These selected individuals are used to update the probability vector by using a Hebbian inspired rule: $p_{l+1}(\boldsymbol{x}) = (1-\alpha)p_l(\boldsymbol{x}) + \alpha \frac{1}{N} \sum_{k=1}^{N} \boldsymbol{x}^l_{k:M}$ where $\alpha \in (0, 1]$ is a parameter of the algorithm. Note that the PBIL algorithm only belongs to the EDA approach if $\alpha = 1$. In this case, PBIL coincides with UMDA. A theoretical study of PBIL could be consulted in González et al. (2000).

5.4.2 Individual representation of learned Bayesian network structures

In order to represent a Bayesian network structure, the representation scheme of the previous experiments is used for the evaluation of approaches (UMDA and PBIL). From the connectivity matrix, two different individual representations could be proposed:

(i) If an order of the variables is given, a node could only be a parent of its following variables within the proposed ordering. The values of the connectivity matrix below the diagonal are zero. The array required to represent a network structure is given by the values of the upper

PBIL
  Obtain an initial probability vector $p_0(\boldsymbol{x})$
  while no convergence **do**
    Using $p_l(\boldsymbol{x})$ obtain M individuals: $\boldsymbol{x}_1^l, \ldots, \boldsymbol{x}_k^l, \ldots, \boldsymbol{x}_M^l$
    Evaluate and rank $\boldsymbol{x}_1^l, \ldots, \boldsymbol{x}_k^l, \ldots, \boldsymbol{x}_M^l$
    Select the N ($N \leq M$) best individuals: $\boldsymbol{x}_{1:M}^l, \ldots, \boldsymbol{x}_{k:M}^l, \ldots, \boldsymbol{x}_{N:M}^l$
    Update the probability vector $p_{l+1}(\boldsymbol{x}) = (p_{l+1}(x_1), \ldots, p_{l+1}(x_n))$
      for $i = 1, \ldots n$ **do**
      $p_{l+1}(x_i) = (1-\alpha)p_l(x_i) + \alpha \frac{1}{N} \sum_{k=1}^{N} x_{i,k:M}^l$
    end while

Fig. 5.18. Pseudocode for the main PBIL algorithm.

triangular connectivity matrix:

$$\mathbf{I} = (c_{12}, \ldots, c_{1n}, c_{23}, \ldots, c_{2n}, \ldots, c_{i(i+1)}, \ldots, c_{in}, \ldots, c_{(n-1)n}).$$

(ii) If all the nodes of the network could be parents of the rest of the nodes, only the values of the c_{ii} elements of the connectivity matrix are zero. An n^2-n dimensional array is required to represent a network structure:

$$\mathbf{I} = (c_{12}, \ldots, c_{1n}, \ldots, c_{i1}, \ldots, c_{i(i-1)}, c_{i(i+1)}, \ldots, c_{in}, \ldots, c_{n1}, \ldots, c_{n(n-1)}).$$

It must be taken into account that the previous arrays represent a directed acyclic graph. Thus, neither genetic crossover and mutation operators of a genetic algorithm, nor the simulation of new individuals in UMDA and PBIL, are closed operations with respect to acyclicity when the ordering is not available: in genetic recombination, and at the simulation step of new individuals of UMDA and PBIL, non-valid individuals could be generated. In this way, a 'repairing' operator is needed to transform not valid individuals (solutions with cycles) into valid ones (directed acyclic graphs). In this work a simple repairing operation is used: once a cycle is detected in the individual, one arc of the cycle is randomly deleted (this is repeated until a directed acyclic graph is achieved).

5.4.3 Experimental results

In order to compare the behaviour of the UMDA and PBIL algorithms and a classic genetic algorithm, a set of experiments are performed with the BIC, K2 and entropy scores. The algorithms are tested algorithms over three databases of the literature:

- The *Asia* Lauritzen and Spiegelhalter, 1988 network is a small Bayesian network that calculates the probability of a patient having tuberculosis,

		BIC		K2		entropy	
	alg.	*score*±*sd*	*gener.*±*sd*	*score*±*sd*	*gener.*±*sd*	*score*±*sd*	*gener.*±*sd*
Ord.	GA	−10947.1±13.6†	21.1±14.35†	−11086.4±387.47†	25.6±16.75†	−1.09±0†	26.1±13.89†
	UMDA	−9890.05±2.51	13.3±1.06	−9802.66±0	15.0±1.76	−1.00±0	14.3±1.34
	PBIL	−9889.26±0	18.5±1.58†	−9802.66±0	19.9±1.28†	−1.00±0	19.7±1.57†
	K2	−9889.26± —	— ± —	−9802.66± —	— ± —	−1.00± —	— ± —
No	GA	−9969.03±46.67†	18.7±3.40	−9813.09±11.41	27.9±9.07	−0.97±0	25.7±6.04
	UMDA	−9917.61±15.29	29.7±6.20†	−9684.36±343.36	34.7±6.68	−0.97±0	24.3±3.13
	PBIL	−9917.46±15.06	30.7±3.89†	−9793.72±39.20	39.5±6.62	−0.97±0	26.8±1.75
	K2	−9968.66±35.57	— ± —	−9804.25±21.28	— ± —	−1.00±0.19	— ± —

Table 5.6. Results of the best scores and the number of evaluations required for the convergence of the *Asia* network. The real values of the BIC, K2 and entropy scores for the network are −9894.16, −9802.66 and −1.00 respectively.

lung cancer or bronchitis, respectively, based on different factors. It has 8 nodes and 8 arcs.

- The *Alarm* Beinlinch et al., 1989 network, as explained in Sections 5.3.3 and 5.2.3, has 37 nodes, 46 arcs and it is related to medical diagnosis.
- The *Water* Jensen et al., 1989 network models the biological processes of a water purification plant. It contains 32 nodes and 88 arcs.

It must be noted that whereas the experimentation with *Alarm* is performed over the first 10000 cases of the well-known database proposed by Cooper and Herskovits, 1992, the experiments with the *Asia* and *Water* networks are carried out over 10000 cases simulated from real Bayesian networks.

Three search techniques (UMDA, PBIL and genetic algorithms) are tested with the same population size, $10n$, where n is the number of variables of the problem (n is 8, 37, and 32 for the *Asia*, *Alarm* and *Water* networks, respectively). The UMDA, PBIL and genetic algorithm general schemes could be modified. In this work, an elitist scheme is used for three search strategies: the new population is formed with the best members of both the previous population and the offspring. The α parameter of PBIL is fixed at $\alpha = 0.5$.

In the case of UMDA and PBIL, half of the best individuals of the populations are selected to form the pool of 'best individuals'. In the case of GA, a rank-based proportional selection is used to select individuals for crossover. Ten independent runs are executed for each combination of score and search technique. When the ordering is taken into account, it is consistent with the topology of the network and it is the same for the ten independent runs.

With the purpose of comparing the obtained results with a 'standard algorithm' to learn Bayesian networks, the results obtained with the well-known K2 algorithm –see Section 4.3– are shown. The K2 algorithm is only executed once the order of the variables is supplied. But when the order is not available, the K2 method is run $10n$ with random orderings.

Tables 5.6, 5.7, and 5.8 show the results obtained for the *Asia*, *Alarm* and *Water* problems, respectively. For each combination of score+search technique, the average score and number of evaluated solutions required for convergence are shown in tables. It is assumed that the search converges when the sum of the scores of the individuals of the previous population is the same

		BIC		K2		entropy	
	alg.	*score*±*sd*	*gener.*±*sd*	*score*±*sd*	*gener.*±*sd*	*score*±*sd*	*gener.*±*sd*
Ord.	GA	−68525.0±1009.9†	106.5±27.24†	−79494.7±1788.1†	155.1±25.68†	−8.59±0.03†	165.8±34.65†
	UMDA	−49430.4±116.40	79.6±5.43†	−47083.1±15.41	73.3±1.77	−6.52±0.02	77.4±3.31†
	PBIL	−49466.2±28.87	66.2±1.69	−47081.6±8.81	77.0±1.94†	−6.52±0.02	74.6±1.07
	K2	−49433.5± —	— ± —	−47103.5± —	— ± —	−6.53± —	— ± —
No	GA	−52331.8±574.21†	174.7±25.84†	−47744.4±289.54†	272.2±31.05†	−6.11±0.08	208.9±36.69†
	UMDA	−51224.3±129.47	149.2±38.02†	−47083.3±10.43	183.8±26.17†	−6.12±0.07	202.5±44.42†
	PBIL	−51250.8±362.90	101.1±7.82	−48271.0±299.81†	117.7±8.06	−6.13±0.04	98.6±5.89
	K2	−52439.7±883.70	— ± —	−49193.4±462.18	— ± —	−6.68±0.15	— ± —

Table 5.7. Results of the best scores and the number of evaluations required for the convergence of the *Alarm* network. The real values of the BIC, K2 and entropy scores for the network are −49687.55, −47086.57 and −6.52 respectively.

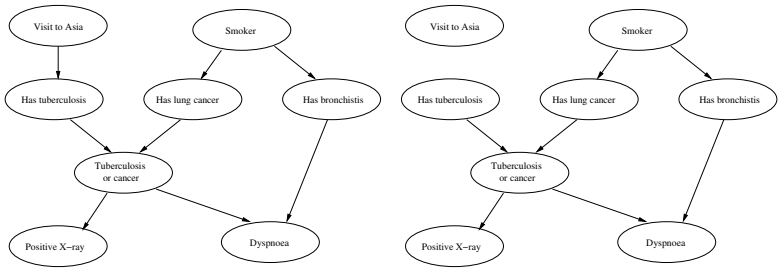
		BIC		K2		entropy	
	alg.	*score*±*sd*	*gener.*±*sd*	*score*±*sd*	*gener.*±*sd*	*score*±*sd*	*gener.*±*sd*
Ord.	GA	−62429.3±586.75†	77.6±26.24	−71175.4±800.08†	124.3±27.40†	−10.34±0.01†	45.44±39.71†
	UMDA	−57119.1±3.11	167.3±10.02†	−56251.4±0	44.6±1.35	−9.79±0	36.2±2.62
	PBIL	−63030.1±1705.3†	178.9±6.71†	−56257.8±8.20	53.9±1.79†	−9.79±0	45.9±1.91†
	K2	−57118.1± —	— ± —	−56251.4± —	— ± —	−9.79± —	— ± —
No	GA	−57967.5±554.27†	91.0±16.92†	−56423.3±100.27	127.0±28.34†	−8.89±0.04	121.9±23.90†
	UMDA	−57141.2±384.66	111.8±21.95†	−56346.9±110.15	155.0±33.86†	−8.87±0.03	112.5±29.26
	PBIL	−57131.9±69.69	65.8±6.37	−56370.4±56.75	72.2±2.86	−8.88±0.02	72.2±4.64
	K2	−57577.7±202.02	— ± —	−56760.0±86.39	— ± —	−9.45±0.19	— ± —

Table 5.8. Results of the best scores and the number of evaluations required for the convergence of the *Water* network. The real values of the BIC, K2 and entropy scores for the network are −120595.94, −56687.60 and −10.07 respectively.

as the sum of the scores of the current population. It must be noted that the objective is the maximisation of the three scores.

A deeper analysis of the results is carried out by means of statistical tests. The Mann-Whitney Mann and Whitney, 1947 test is performed to determine the significance of the differences shown in the score and in the number of evaluated solutions for each scoring function and individual representation. For each scoring measure, statistically significant differences with respect to the algorithm with the best average score obtained by each individual representation are noted in Tables 5.6, 5.7 and 5.8; the same test is carried out in relation to the algorithm with the lowest number of evaluated solutions required for convergence. The symbol † denotes a statistically significant difference with respect to the best search algorithm at the 0.05 confidence level in Tables 5.6, 5.7 and 5.8.

For the *Asia*, *Alarm* and *Water* networks, when the ordering is supplied, the UMDA and PBIL algorithms obtain competitive results with respect to GA with the lowest number of generations. The results of UMDA and PBIL improve the real values of the networks and the value of the network learned by the K2 algorithm, except for *Water* with the BIC score. The number of generations required for convergence by PBIL and UMDA is, in all cases, lower than the number of generations required by GA, except for *Water* with the BIC score.



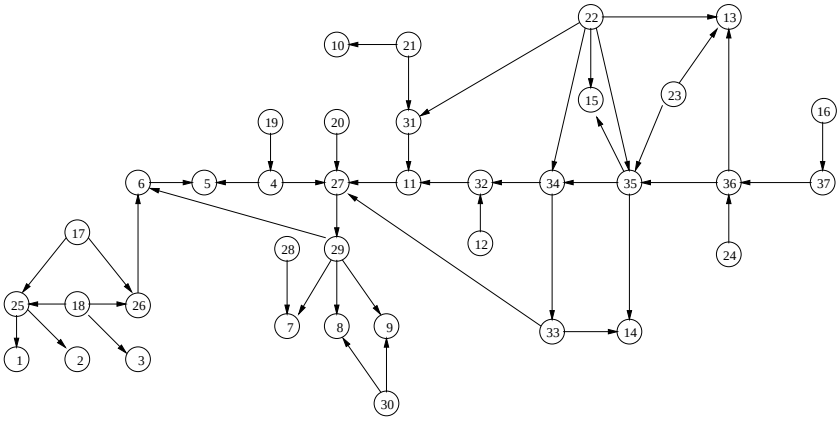
(a) *Asia* structure (b) Network structure learned by UMDA

Fig. 5.19. The original *Asia* network structure has 8 nodes and 8 arcs. The network structure learned by UMDA and PBIL with BIC and entropy metrics has 7 arcs. It can been seen that only one arc is missing: the one from the *Visit to Asia* node to *Has tuberculosis* node.

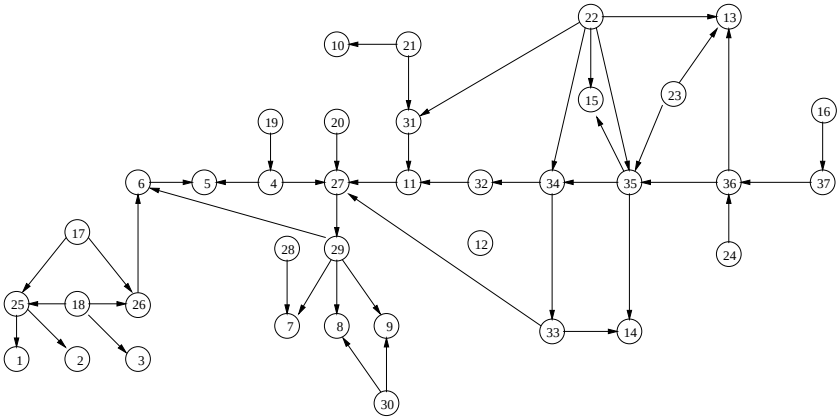
When the ordering is not taken into account, it must be noted that the results of GA are competitive, but UMDA always obtains the best results (except for *Alarm* with the entropy score). These results improve those obtained by the K2 algorithm with random orders. PBIL needs the lowest number of generations in all cases, obtaining score values not significantly different to those obtained by UMDA.

It must be noted that, in all cases, for the three metrics and the three networks, GA obtains better results if the ordering is ignored, that is, it obtains better results when the problem is more complex than if the ordering is taken into account. This could mean that the stopping criterion is restrictive to GA, and the algorithm stops when the population is not uniform. Figure 5.22 shows that when UMDA and PBIL stop improving GA still improves slowly. GA could possibly obtain better results with other less restrictive stopping criteria when the ordering is available.

Comparisons between the structure of the networks with the best score values and the original networks are also performed. Three types of differences are measured with respect to the original network: the Hamming distance, the number of extra arcs and the number of missing arcs in the network learned. The networks more similar to the original ones are obtained when ordering is taken into account. In Figures 5.19, 5.20 and 5.21, the networks closest to the original ones are shown. It must be noted that the UMDA and PBIL algorithms with the K2 score obtain the original *Asia* network. In Figure 5.19 the network structure drawn is the one obtained by UMDA and PBIL with BIC and entropy scores. In the case of the *Alarm* network, the structure depicted in Figure 5.20 is obtained with the three metrics by the UMDA algorithm. In the case of the *Water* network, the structure shown in Figure 5.21 is the most common structure among the set of structures obtained by the three algorithms, and it is obtained by the UMDA algorithm using the K2



(a) *Alarm* structure



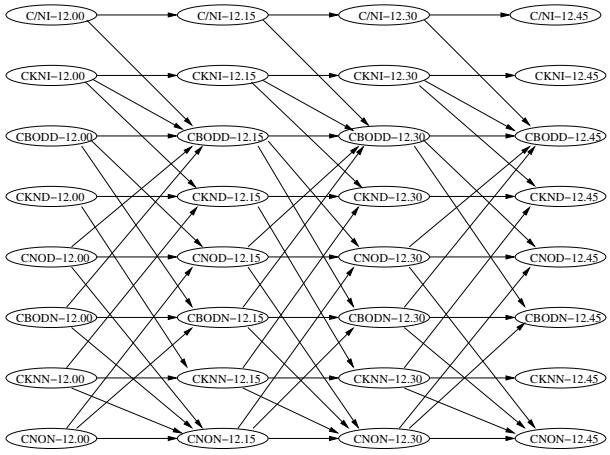
(b) Network structure learned by UMDA

Fig. 5.20. The original *Alarm* network structure has 37 nodes and 46 arcs. The network structure learned by UMDA with the three metrics has 45 arcs. It could been seen that the arc from node 12 to node 32 is missing.

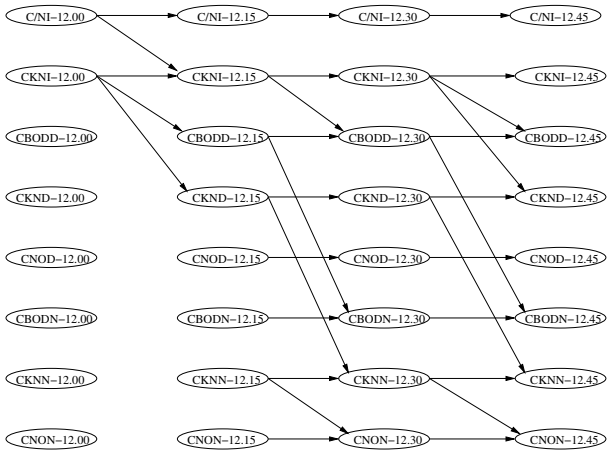
metric. If ordering is ignored, the structures learned are very different from to the original network.

Figures 5.22 and 5.23 show the evolution of the best values found during the search process with respect to the number of evaluations in the search process in a typical run of the *Alarm* network. In Figure 5.22 the ordering is available, and it is ignored in Figure 5.23.

Figure 5.22 shows how UMDA and PBIL obtain a considerable improvement in the first 10000 evaluations. In further evaluations, this gain is maintained. It is assumed that the best values found by UMDA and PBIL increase logarithmically. GA seems to increase logarithmically as well, but the growth



(a) *Water* structure



(b) Network structure learned by UMDA

Fig. 5.21. The original *Water* network structure has 32 nodes and 66 arcs. The network structure learned by UMDA with the K2 metric has 36 missing arcs.

is small and slow with respect to UMDA and PBIL. The number of generations required by GA is higher than those needed by UMDA and PBIL.

In Figure 5.22, it can be seen that the growth of the best values is very similar among the three search algorithms. It must be noted that UMDA and PBIL need fewer evaluations than GA. It seems that UMDA and PBIL find better values before GA in the search process, maintaining this difference in the rest of the search.

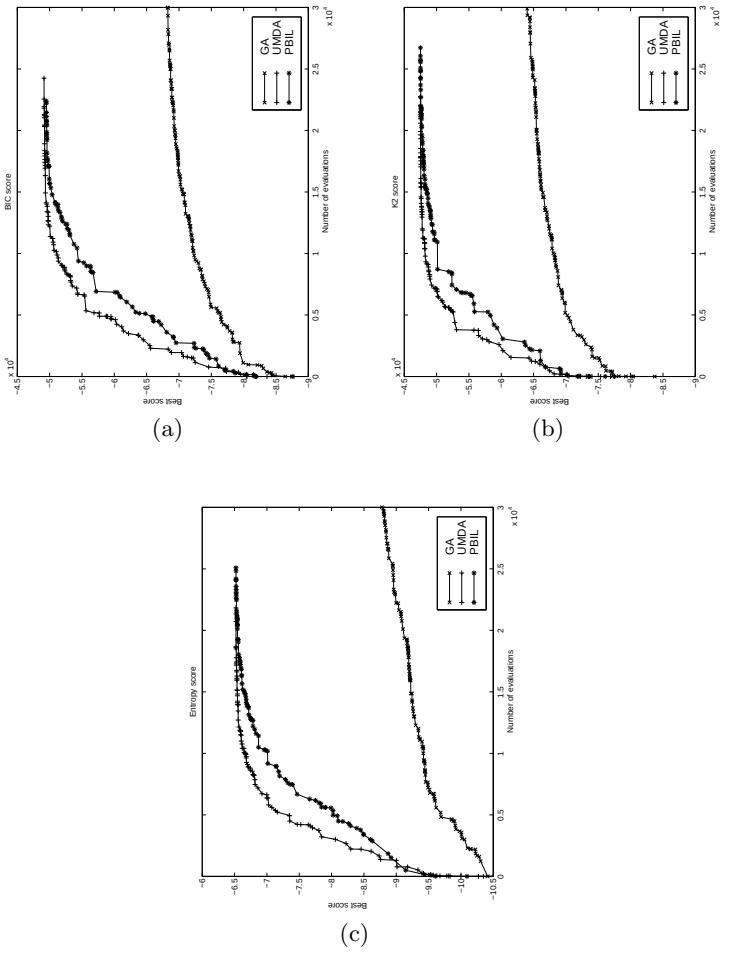


Fig. 5.22. Evolution of the best value found in the search process for the *Alarm* network when ordering is available with (a) BIC score, (b) K2 score and (c) entropy score.

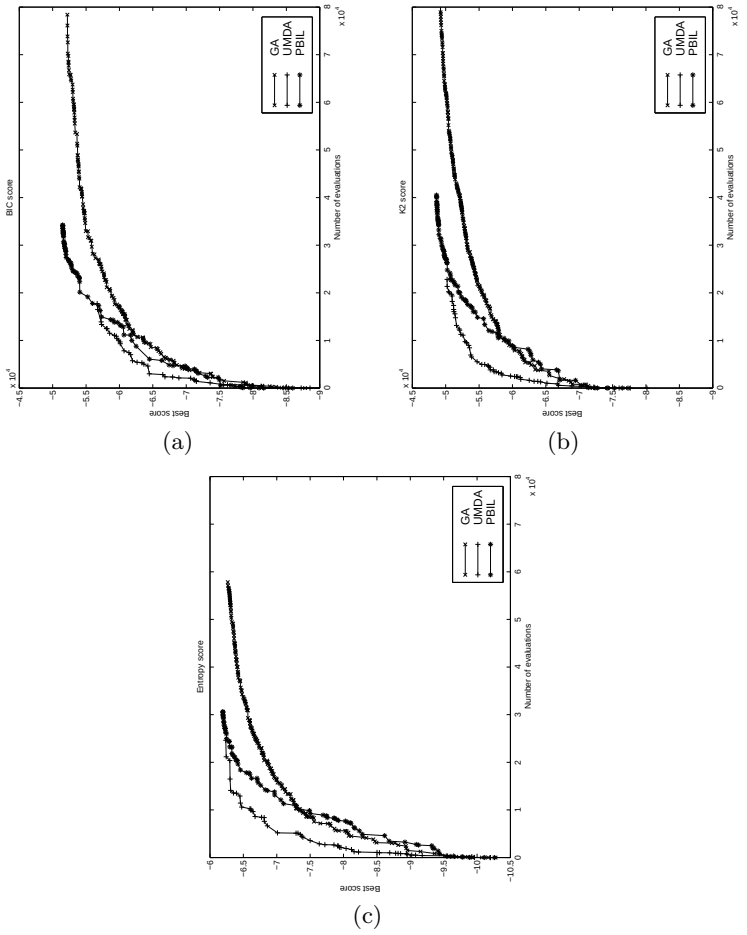


Fig. 5.23. Evolution of the best value found in the search process for the *Alarm* network when ordering is ignored with (a) BIC score, (b) K2 score and (c) entropy score.

6

Conclusions and future work

6.1 Conclusions

In this part, the learning of Bayesian networks is analysed from the point of view of optimisation paradigm. Learning a Bayesian network could be seen as an optimisation problem based on three basic components: a scoring function, a search space and a search engine.

The aim of this part is two-fold: the revision of the score+search approach to learning Bayesian networks and, especially, the introduction of three optimization techniques, which have been never used to perform this task. Although the scoring functions applied (K2, BIC and entropy) and the search space used (the space of directed acyclic graphs represented by a connectivity matrix) are well-known and widely used in the literature, the main contribution in this field is the proposal of three novel search engines to perform the search process: floating methods, GRASP and EDAs.

Floating methods have been proposed in the feature subset selection area with the aim of avoiding the nesting problem and reconsidering each decision taken. These algorithms are adapted to learn Bayesian networks where the inclusion and exclusion of arcs are the decisions to be taken and reconsidered.

The GRASP procedure has been proposed in the optimisation area. An iterative process is performed to construct a solution in a semi-greedy way with a local search to improve the result. In order to perform the learning of Bayesian networks, the construction step is similar to the B algorithm but with a random element, and the local search is performed by the B3 algorithm.

Finally, the EDAs are population-based algorithms proposed to overcome some of the difficulties of genetic algorithms. Once the individual representation is set, any optimisation problem could be run. In this work, two different individual representations are proposed depending on whether an ordering among the domain variables is taken into account.

The empirical study is carried out in different ways depending on the intrinsic characteristic of each search engine proposed. The empirical study is

based on the differences among the scoring values attained and the number of generations to convergence.

Although the numerical results of the novel search engines proposed (floating methods, GRASP and EDAs) are not as good as expected, they are competitive with respect to the standard 'benchmark' methods (B, B3, K2 and genetic algorithms) carried out to perform a comparison. In spite of the extra computational cost related to the novel methods proposed, it does not provide a statistically significant improvement in the scoring measure value. However, the scheme followed in the experimentation is a basic and general scheme of these search algorithms. An adequate tuning of the parameters and the choices made to perform the search could enhance the numerical results. Moreover, a set of improvements has been proposed for floating methods Somol et al., 1999 and GRASP Resende and Ribeiro, 2003, which are not applied in this work.

Furthermore, a qualitative study of the network structures learned is performed. It must be noted that the Bayesian network structures learned are similar to the original Bayesian networks when they contain a 'reasonable' number of nodes and arcs.

6.2 Future work

Many adaptations to and improvements on this work could be performed in the future. Bearing in mind the three basic components of the optimisation process, several enhancements could be carried out. In Section 4.3, these elements are studied in the field of learning Bayesian networks. Thus, other combinations of scoring functions, search spaces and search engines could be developed.

- Scoring functions. The use of three other well-known scoring functions of the literature such as AIC, BD and MDL. Nevertheless, the scoring functions are usually maintained in order to compare with the results in the literature.
- Search spaces. The space of structures has limitations reported in the literature Anderson et al., 1997. These limitations are overcomed by other search spaces.
- Search engines. Several search algorithms could be proposed to perform the search of Bayesian network structures. Search methods like ant colony optimisation de Campos et al., 2002 or tabu-search Bouckaert, 1995; Munteanu and Cau, 2000 have demonstrated their capacity to learning Bayesian networks.

However, the use of floating methods, GRASP and EDAs over the space of equivalence classes is an interesting future work line to explore the power of these search algorithms.

Part III

Supervised Classification by Bayesian Networks

7

Introduction

The supervised classification task involves the labelling of unlabelled cases. Due to the growth of the database size in the last decade, this task can be performed automatically. Nevertheless, classifier performance decreases when variables unrelated to class or redundant in relation to other variables are included in the model. Hence, a previous step to reject unrelated variables is required to improve classifier accuracy. In this chapter, the supervised classification problem, feature subset selection and accuracy estimation are presented. This way, the foundations related to the Bayesian classifiers presented in next sections are set.

The chapter is organised as follows. In Section 7.1, the supervised classification problem is introduced. Section 7.2 presents the feature subset selection task. The chapter concludes with Section 7.3, where methods to estimate the accuracy of a classifier are presented.

7.1 Supervised classification problem

The artificial intelligence subarea known as *classification* consists of *supervised classification* and *unsupervised classification* tasks. The aim of unsupervised classification is to discover the underlying structure of a given dataset.

The *supervised classification* problem is defined as set a class value of the r_0 values of the C variable to an instance vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \mathcal{R}^n$. The real class value is denoted by $c \in \{1, 2, \ldots r_0\}$. A classifier could be seen as a function, γ, which labels instances:

$$\gamma : (x_1, x_2, \ldots, x_n) \rightarrow \{1, 2 \ldots, r_0\}$$

Intuitively, the classification problem could be seen as the following: given a labelled instance dataset, build (by means of a learning algorithm or inducer) a black box (classifier) to predict the label of new unseen instances.

Several families of classifiers have been developed in the past years. First, the *statistic based* paradigms are influenced by statistics. The development

of these classifiers is related to the maximisation of the likelihood or conditional likelihood of the data given the classifier. Prior hypotheses are made concerning data distribution and the free parameters of the classifier for the induction process. Discriminant analysis Fisher, 1936 and logistic regression Hosmer and Lemeshow, 1989 are classification models which belong to this family.

However, during the 80s and 90s, classifier evolution has accelerated due to the increase in dataset size and number. The classification methods proposed for the use of large datasets are *machine learning* techniques. Classifier induction is automatically guided by the data. Thus, emphasis on the prior hypotheses of the statistic based paradigms is lost.

These explosion in the number of classification models makes feasible an organisation of classifiers into families. Several families of classifiers could be set, some of the most popular classifiers belong to one of the following:

- *Decision trees.* They label instances by sorting them throughout the tree, from the root to a leaf node. Each inner node corresponds to a variable and an arc (or branch) reaching a child represents a possible value of that variable. A leaf represents the predicted value of the class variable given the values of the variables represented by the path from the root. ID3 Quinlan, 1986 and C4.5 Quinlan, 1993 are popular decision tree inducers.
- *Decision rules.* A set of ordered IF-THEN decision rules are induced. If the condition of the rule is matched, the class label appears in the THEN part. To label new examples, each rule is executed following the ordering until one is matched. CN2 Clark and Nibblet, 1989 and Ripper Cohen, 1995 are decision rule inductors.
- *Instance based* classifiers. The main characteristic of this family is that each sample of the train data is maintained in the data structure. Inducers such as IB1 Aha et al., 1991 and PEBLS Cost and Salzberg, 1993 are instance based methods.
- *Neural networks.* A massively parallel collection of small and simple processing units. A typical feed-forward neural network consists of a set of nodes: *input* nodes, *hidden* nodes and *output* nodes. Hence, for classification purposes, the predicted class is an output node. To classify a new instance, a value is applied to each input node. The Boltzmann machines Ackely et al., 1985 and the self-organising map Kohonen, 1995 are methods based on neural networks.

This division into families is neither intensive nor exhaustive. Hence, there are other types of classifiers. It must be noted that this work mainly focusses on Bayesian classifiers, which can be considered as another family of classifiers. They are based on Bayesian networks and are thoroughly studied in the next chapters.

In the classification task with two class values, given a classifier and an instance, four feasible outcomes can be produced: *true positive* if the instance is positive and labelled as positive, *true negative* if the instance is negative

		true class	
		true	false
hypothesised class	true	*true positives*	*false positives*
	false	*false negatives*	*true negatives*

Table 7.1. Confusion matrix.

and labelled as negative, *false positive* if the instance is negative but labelled as positive and, finally, *false negative* if the instance is positive labelled as negative. Thus, given a classifier and a set of instances, a *confusion matrix* or *contingency table* can be built representing the four outcomes of the set of instances. Figure 7.1 shows the scheme of a confusion matrix.

Some scores can be calculated from a confusion matrix. The numbers in the major diagonal represent the correct classification of instances. The number off this diagonal represents the misclassification error. The *sensitivity* of a classifier is estimated by means of the confusion matrix as:

$$sensitivity = \frac{true\ positives}{true\ positives + false\ negatives}$$

On the other hand, *specificity* is related to the negative instances and is estimated as:

$$specificity = \frac{true\ negatives}{true\ negatives + false\ positives}$$

Apart from the confusion matrix, any classifier has a related *misclassification cost matrix*, $cos(r, s)$ where r is the real label value of the instance to classify and s is the assigned label value. In the special case of *loss function 0/1*, the cost matrix is:

$$cos(r, s) = \begin{cases} 1 \text{ if } r \neq s \\ 0 \text{ otherwise} \end{cases}$$

Assuming the existence of a joint probability distribution,

$$p(x_1, x_2, \ldots, x_n, c) = p(c|x_1, x_2, \ldots, x_n)p(x_1, x_2, \ldots, x_n),$$

and $cos(r, s)$, a Bayesian classifier that minimises the total misclassification cost could be built Duda and Hart, 1973 using the following assignment:

$$\gamma(\boldsymbol{x}) = arg \min_k \sum_{c=1}^{r_0} cos(k, c)p(c|x_1, x_2, \ldots, x_n)$$

In the case of loss function 0/1, the Bayesian classifier assigns the *most a posteriori probable class* to a given instance, that is:

$$\gamma(\boldsymbol{x}) = arg\max_{c} p(c|x_1, x_2, \ldots, x_n) = arg\max_{c} p(x_1, x_2, \ldots, x_n, c)$$

In real problem domains, the joint probability distribution $p(x_1, x_2, \ldots, x_n, c)$ is unknown. It could be estimated using a simple random sample $\left\{(\boldsymbol{x}^{(1)}, c^{(1)}), (\boldsymbol{x}^{(2)}, c^{(2)}), \ldots, (\boldsymbol{x}^{(N)}, c^{(N)})\right\}$ obtained from the joint probability distribution.

Roughly speaking, the classifiers could be divided into two main types Rubinsteins and Hastie, 1997:

- Generative classifiers
  Instead of directly estimating of the a posteriori distribution of the class $p(c|x_1, x_2, \ldots, x_n)$, the class conditioned densities $p(x_1, x_2, \ldots, x_n|c)$ and priors $p(c)$ are estimated. Applying the Bayes rule, $p(c|x_1, x_2, \ldots, x_n)$ is obtained as follows:

$$p(c|x_1, x_2, \ldots, x_n) = \frac{p(x_1, x_2, \ldots, x_n|c)p(c)}{\sum_{c=1}^{r_0} p(x_1, x_2, \ldots, x_n|c)p(c)}.$$

  The parameters of the classification model are estimated by means of their maximum likelihood parameter estimations. It is necessary to maximise the logarithm of the joint likelihood to reach the maximum likelihood parameter estimations:

$$\begin{aligned}\mathcal{L}\left((\boldsymbol{x}^{(1)}, c^{(1)}), (\boldsymbol{x}^{(2)}, c^{(2)}), \ldots, (\boldsymbol{x}^{(N)}, c^{(N)})\right) &= \log \prod_{j=1}^{N} p(\boldsymbol{x}^{(j)}, c^{(j)}) \\ &= \sum_{j=1}^{N} \log p(\boldsymbol{x}^{(j)}, c^{(j)}).\end{aligned}$$

  There are examples of informative classifiers in the literature, for instance in Duda and Hart (1973) and Fisher (1936).
- Discriminative classifiers
  The discriminative approach models directly estimate the a posteriori distribution of the class by means of the predictive variables, that is, $p(c|x_1, x_2, \ldots, x_n)$. Figure 7.1 shows this approach.
  In this case, parameter estimation is carried out by maximising the logarithm of conditioned likelihood, that is:

$$\begin{aligned}\mathcal{L}\left((c^{(1)}|\boldsymbol{x}^{(1)}), (c^{(2)}|\boldsymbol{x}^{(2)}), \ldots, (c^{(N)}|\boldsymbol{x}^{(N)})\right) &= \log \prod_{j=1}^{N} p(c^{(j)}|\boldsymbol{x}^{(j)}) \\ &= \sum_{j=1}^{N} \log p(c^{(j)}|\boldsymbol{x}^{(j)}).\end{aligned}$$

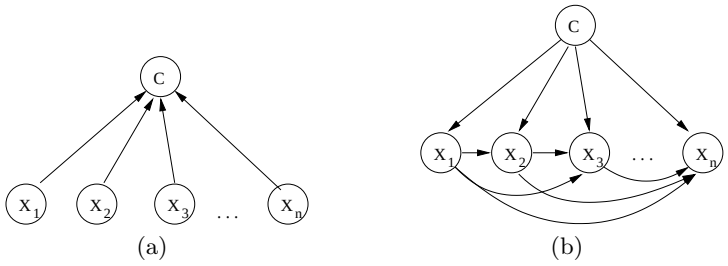


Fig. 7.1. (a) Bayesian network represents $p(C, X_1, X_2, \ldots, X_n)$. (b) Bayesian network represents $p(C, X_1, X_2, \ldots, X_n)$ after changing the arc direction.

This parameter estimation is usually harder to carry out due to the need for iterative procedures to solve it.
Logistic regression Hosmer and Lemeshow, 1989 and generalised additive models Hastie and Tibshirani, 1990 are discriminative classifiers.

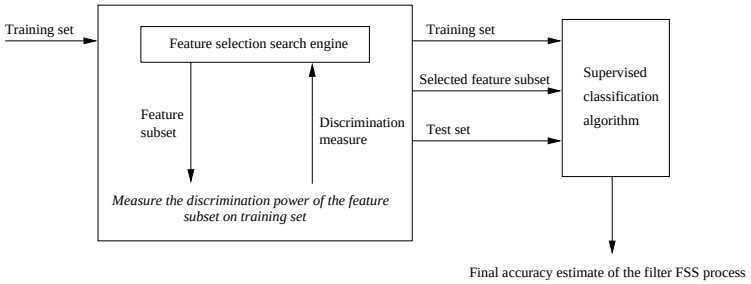
7.2 Feature subset selection problem

Although the *feature subset selection* problem has long been studied, a unique, general and concise definition has not been reached yet. Different definitions can be found in the literature Kohavi and John, 1997; Tsamardinos and Aliferis, 2003; Weston et al., 2000.

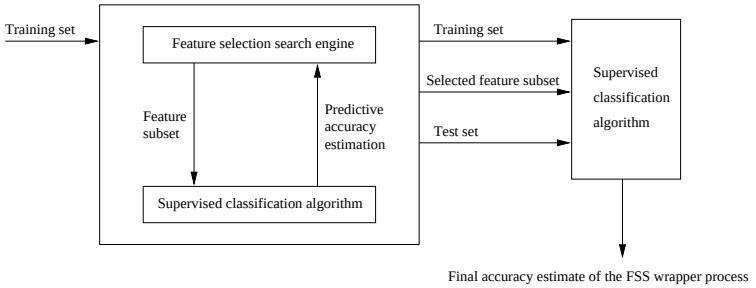
In spite of the several definitions of the problem, the feature selection problem may be defined as the following question: are all the problem variables useful for classification? Therefore, it is a crucial task in the supervised classification problem. The feature selection task reduces the number of variables of a problem domain. In order to determine the 'useless' variables, the *redundancy* and *irrelevancy* concepts become crucial terms. The *irrelevant* features do not affect the underlying structure of the data in any way. The *redundant* features, given a subset of variables, do not provide anything new when describing the underlying structure. Both, the irrelevant and redundant variables could decrease the performance of the classifier.

These are the main advantages of feature selection:

- An improvement in accuracy and computational time: parsimonious models require less computational time to perform inference and prediction.
- A cost reduction due to data acquisition reduction: unnecessary variables could be expensive to observe. Moreover, in medical domains, unnecessary medical tests could be dangerous for the patients.
- An improvement in the comprehensibility of the model: when the learning method is carried out to understand the underlying domain process better, unnecessary variables make understanding difficult . Feature reduction provides less complex and more understandable models for domain experts.



(a) Filter approach



(b) Wrapper approach

Fig. 7.2. General schemes for feature reduction: (a) *filter* and (b) *wrapper* approaches.

Usually, the main goal of feature selection is an accuracy improvement. Then, an evaluation function measuring the quality of the feature subset is required. Traditionally, functions that measure the power to distinguish between class values by looking only at the intrinsic characteristics of the data have been used. Functions like log-likelihood of the data and information based measures (like conditional entropy of the class or mutual information) have been largely carried out to perform feature reduction Blum and Langley, 1997; Koller and Sahami, 1996; Lewis, 1992; Singh and Provan, 1996; Tourassi et al., 2001; Zaffalon and Hutter, 2002. These of scores are known as *filter functions* and the feature selection process based on them is known as *filter approach*.

However, John et al. (1994) reports that, when the goal is to maximise the accuracy of the classifier, feature selection should depend not only on the data and the concept to learn, but also on the characteristics of the classifier. Although previous works report the use of the misclassification error to guide the feature selection process Siedelecky and Sklansky, 1988; Stearns, 1976, it is assumed that the *wrapper approach* appears in the 90s due to the previous computational limitations. In the wrapper approach, the classification algorithm is therefore used as a black box in order to estimate the quality of the feature selection. When a feature subset is selected by the search algorithm, its predictive accuracy –estimated using the supervised classification algorithm

proposed to generate the final model– is considered a search process guide. Differences between the *filter* and *wrapper* approach are displayed in Figure 7.2.

In order to solve a feature selection problem, an objective function measure must be fixed. When the feature selection is carried out with classification purposes, the maximisation of the classification model's accuracy is the objective. The feature selection process could be tackled as a usual search problem where each possible feature subset is a problem solution.

From this point of view, several search strategies have been proposed in the literature to perform feature reduction. The set of search strategies could be divided into two main groups: *exhaustive* and *heuristic* algorithms. The depth-first Liu and Motoda, 1998, breadth-first Liu and Motoda, 1998 and branch and bound Chen, 2003; Narendra and Fukunaga, 1977; Somol et al., 2004 are exhaustive methods which have been proposed as search algorithms for feature selection. Although exhaustive methods guarantee the optimum solution, they can only be used with monotone scoring functions. However, the functions to be optimised in real domains are seldom monotone. Heuristic methods have long been used due to the former reason and to the computational cost related to the exhaustive search as the number of variables raises.

The heuristic methods could be divided into *deterministic* and *non-deterministic* algorithms. In the deterministic subgroup, sequential forward selection and sequential backward elimination Kittler, 1978 become classic algorithms for feature reduction. Then, the best-first Kohavi and John, 1997 and floating selection algorithms Pudil et al., 1994; Somol et al., 1999 are proposed. These deterministic algorithms, given the same initial conditions, attain the same final solution, but not always the optimum. To avoid the local optima, the non-deterministic search strategies use randomness. Non-deterministic algorithms like simulated annealing Debuse and Rayward-Smith, 1999; Doak, 1992, genetic algorithms Guerra-Salcedo and Whitley, 1998; Yang and Honavar, 1998 and estimation of distribution algorithms Cantú-Paz, 2002; Inza et al., 2001a have been proposed for feature reduction. Liu and Motoda (1998) compiles other types of classifications of feature selection methods.

7.3 Accuracy estimation: measuring the quality of the classification model

The *accuracy* of a classifier is an estimation of the probability of correctly classifying a randomly selected instance Kohavi, 1995. Hence, accuracy estimation is a crucial task when facing supervised classification problems. Although there is an open discussion about which measure is the 'best' one to select a classifier for a problem domain, accuracy is often used as a comparison function. Therefore, estimating the accuracy of a classifier is important not only to predict the future behaviour but also to select a classifier from a given set Wolpert, 1992. Moreover, in domains like medical diagnosis, it is

important to acceptably assess either the sensibility or the specificity of the classification model.

The accuracy estimation is straightforward, with a low uncertainty when the data is large. Nevertheless, accuracy estimation is performed with a limited number of cases. Thus, some difficulties must be taken into account Mitchell, 1997:

- In testing the accuracy of a classifier, its error rate estimate tends to be *biased* if it is assessed from the same set of examples that was used to construct the classification model. This is especially critical when the classification algorithm considers a very large space of possible models, enabling it to overfit the training cases. To obtain an unbiased accuracy estimation, the classifier should be tested on a set of examples chosen independently of the examples that build it.
- The accuracy estimation could vary from the true accuracy, depending on the specific makeup of the particular test examples. This is especially critical when a small number of test examples is provided: in this case, the error rate estimation tends to have a large *variance*.

Several methods to estimate accuracy have been proposed in the literature showing their advantages with respect to the *bias* and *variance* concepts. Methods like bootstrapping Efron and Tibshirani, 1993 and jackknife Rao and Shao, 1992 have been successfully applied in different areas. More recently, Nadeau and Bengio (2003) reviews some of these methods and proposes a new way to calculate their *variance*.

Historically, two classic methods for accuracy estimation have been used: the *resubstitution estimate* and the *holdout*.

- The *resubstitution estimate* or apparent error estimates the accuracy of the classifier by testing the same data used to induce it.
- The *holdout* or test-sample estimation randomly splits the data into two mutually exclusive datasets. Usually 2/3 of the data is used as a training set and 1/3 as a test set. Then, the accuracy of the classifier induced with the training data and tested with the test data is used as an accuracy estimation of the classifier learned with all the data.

Although accuracy estimation methods has been criticised for the past years and several accuracy estimation algorithms have been proposed Provost et al., 1998; Nadeau and Bengio, 2003; Ng, 1997, it is still commonly used.

7.3.1 *k*-fold cross-validation

In *k-fold cross-validation* Stone, 1974, the dataset is randomly split in k mutually exclusive subsets or folds of the same size approximately. The inducer is performed over $k-1$ folds and tested over the remaining fold. This process is repeated k times with a different remaining fold each. Then, the k-fold cross-validation estimates the accuracy by averaging out the accuracy of these k

folds. The standard deviation could be reported in the same way. The variance of the cross-validation estimation should be approximately the same, independently of the number of folds Kohavi, 1995. Although any k ($1 < k \leq N$), where N is the number of instances of the dataset, is feasible, the most common value in the literature is $k = 10$.

Methods to improve the estimation of k-fold cross-validation have been proposed. *Stratified cross-validation* Breiman et al., 1984 stratifies the folds in such a way that they contain approximately the same proportion of classes as the original dataset.

7.3.2 *Leave-one-out* cross-validation

Leave-one-out cross-validation Lachenbruch and Mickey, 1968 is a particular case of k-fold cross-validation. In this cross-validation method, k is fixed at N, where N is the number of instances of the dataset. Thus, the learning algorithm is performed over $N - 1$ instances and tested over the remaining example.

The main advantage of leave-one-out cross-validation is that it is almost unbiased, but it has higher variance, which sometimes leads it to unreliable estimates Efron, 1983. Despite this fact, it is usually applied when the dataset size is relatively small.

8

Bayesian classification models

Bayesian classification models emerge from the use of Bayesian networks for classification purposes. This field has been used in the machine learning area with promising results. Depending on the restrictions imposed on the Bayesian network structures, different classifiers have been proposed. Several structures for Bayesian classification models presented in the literature are reviewed in this work. Nevertheless, this revision is not exhaustive and it is restricted to naive Bayes, seminaive Bayes, tree augmented naive Bayes and k dependence Bayesian classifier. Bayesian classification models like general Bayesian networks and Bayesian multinets are not included. Consult Larrañaga (2003) for details about these Bayesian classification models.

The chapter is arranged as follows. The next section briefly introduces the Bayesian classifiers. Afterwards, in consecutive sections, naive Bayes, seminaive Bayes, tree augmented naive Bayes and k dependence Bayesian classifiers are presented.

8.1 Introduction

For the past years, probabilistic graphical models in combination with the classification task have experienced an important growth. The graphical representation of Bayesian classifiers is intuitive, allowing the domain experts to understand the underlying probabilistic classification process without a deep knowledge of Bayesian classifiers.

A model hierarchy of increasing complexity could be established for Bayesian classifiers, where the naive Bayes is at the bottom and a general Bayesian network is at the top of this hierarchy. Nevertheless, this work focusses on less complex Bayesian classifiers: naive Bayes, seminaive Bayes, tree augmented naive Bayes and k-dependence Bayesian classifiers.

The restrictions imposed on naive Bayes, selective naive Bayes, seminaive Bayes, tree augmented naive Bayes and k-dependence Bayesian classifiers are due to the type of relations between variables that they consider. In spite of

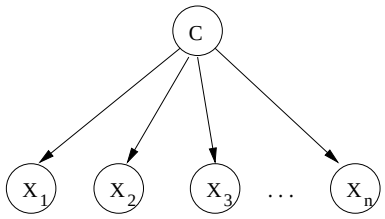


Fig. 8.1. Structure of a naive Bayes model.

their limitations, these Bayesian classifiers provide a set of properties that can be appreciated by domain experts. Their graphical structure facilitates interpretability and understanding, reflecting probabilistic relationships between domain variables. The conditional and marginal distributions of the model could be of interest to understand the uncertainty of the analysed domain better. Another interesting characteristic is that, when computational time is a critical factor, these Bayesian classifiers are quickly learned from a database by means of the original inductors. Furthermore, once the Bayesian classification model is induced, it is able to quickly obtain a prediction for an unseen example and add the knowledge of this unseen example to the model.

8.2 Naive Bayes

Naive Bayes Minsky, 1961 is the simplest Bayesian classification model. It is built upon the assumption of conditional independence of the predictive variables given the class. Although this assumption is violated in numerous occasions in real domains, the paradigm still performs well in many situations Domingos and Pazzani, 1997; Hand and You, 2001. Making this assumption, the prediction of the class for an unseen instance is simplified. In Figure 8.1, a graphical representation of the structure of a naive Bayes is shown.

Although naive Bayes has very often been used in pattern recognition Duda and Hart, 1973, the first time that naive Bayes appears in machine learning literature was at the end of the 80s Cestnik et al., 1987. In Cestnik et al. (1987), naive Bayes accuracy is compared with the accuracy of other more 'sophisticated' predictors. Gradually, the naive Bayes classifier has been used by machine learning researchers due to its robustness in supervised classification tasks.

In spite of its wide use, there is not only one name to refer to it. Names like *idiot Bayes* Hand and You, 2001; Ohmann et al., 1988, *naïve Bayes* Kononenko, 1990, *simple Bayesian classifier* Domingos and Pazzani, 1997; John, 1997 and *independent Bayes* Crichton et al., 1998; Todd and Stamper, 1994 can be found in the literature.

Naive Bayes has been used in the machine learning field for years. Several applications have been presented in different areas like: medicine, insurance,

industry... In most cases, naive Bayes attains promising results and is competitive with respect to other sophisticated classifiers. In the literature, papers which support this fact can be found. For instance, in the medical domain, in Titterington et al. (1981) the best results are obtained with the naive Bayes classifier over brain damage data. In Mani et al. (1997), as in the former paper, naive Bayes is the best inductor in a database of breast cancer recurrence. Others medical areas have successfully dealt with naive Bayes: liver diseases Croft and Machol, 1987, dyspepsia Fox et al., 1980, abdominal pain Gammerman and Thatcher, 1991, thyroid diseases Nordyke et al., 1971 and heart diseases Russek et al., 1983. However, in other papers, the naive Bayes does not work as well as expected King et al., 1995; Michie et al., 1994; Heckerman and Nathwani, 1992.

Following the review of the naive Bayes medical applications, Bailey (1964) also uses naive Bayes in medical diagnosis, proposing some ideas to add independences among the variables. In the same way, Boyle et al. (1966) uses the naive Bayes paradigm to distinguish between three types of goitre. In order to perform a comparison, Fryback (1978) presents the results of two models: naive Bayes and a model which allows dependencies between pairs of variables. It reports that if the data violates the independence condition, the more the variables included, the worse the results obtained with naive Bayes.

An interesting paradigm is the classification tree inducer called *Assistant* Kononenko, 1990. In this paper, a thorough comparison between *Assistant*, naive Bayes and the average prognosis of four medical experts is carried out. Four medical problems are taken into account: primary tumour location, recidivistic breast cancer, thyroid disease and reumatology. Interestingly, the medical experts attain worse accuracy results than *Assistant* and naive Bayes. Moreover, naive Bayes obtains higher accuracy than *Assistant*.

Movellan et al. (2002) applies naive Bayes to a neurology database. Cell response to several stimuli and contexts whose effects are considered independent is studied. Although naive Bayes usually outperforms, when compared with respect to several induction techniques, in the acute abdominal pain domain Ohmann et al., 1996, the results attained do not show statistically significant differences between the inducers proposed.

More literature of naive Bayes in medical domains can be found in Adams et al. (1986), Brunk et al. (1975), Coomans et al. (1983), Cornfield et al. (1973), de Dombal (1991), duBolay et al. (1977), Edwards and Davies (1984), Heckerman et al. (1992), Spiegelhalter and Knill-Jones (1984), van Woerkom and Brodman (1961) and Warner et al. (1961).

Although naive Bayes is a very extended paradigm in medical domains, it has also been applied many other new domains. For instance, in the web page domain, Pazzani et al. (1996) presents an agent called *Syskill & Webert*. Once the user profile is learnt, the user's interest in several web pages being stored, it is able to show links that could be of interest to the user. In order to develop the *Syskill & Webert* core classifier, five machine learning paradigms are compared: naive Bayes, K-NN, ID3, perceptron and a multi-layer neural

network trained with backpropagation. Finally, naive Bayes is the classifier selected for *Syskill & Webert* due to its speed in learning and prediction. Moreover, naive Bayes provides a ranking among the class values, that is, among the new links to explore.

An application in a similar domain is proposed by Miyahara and Pazzani (2000). The paper presents a collaborative filter approach based on naive Bayes. By means of collaborative filtering, new items (CDs, books, movies ...) of interest to particular users are recommended according to other users' opinions. Two proposed versions of the naive Bayes model empirically outperform the classical approach to the problem based on the Pearson correlation coefficient.

Text classification has become an interesting domain in the last years due to the increase in on-line documents. Hence, several applications of machine learning techniques have been presented. Naive Bayes is proposed in McCallum and Nigam (1998), where a comparison of two different models (Bernouilli and multinomial) for text classification based on naive Bayes is carried out. When the Bernouilli model is taken into account, a binary vector represents a text. Each vector component is a text word and the component value shows whether the word is present in the text. On the other hand, when the multinomial model is used, the characteristic vector stores the number of appearances of the word in the text. Naive Bayes obtains higher accuracy when the multinomial model is applied.

Another study applies naive Bayes to a hard disk failure problem Hamerly and Elkan, 2001, by means of eleven internal measures of the driver. A real database of 1936 drivers is used to compare naive Bayes with the usual industrial techniques. The study empirically shows that naive Bayes obtains better performance.

Due to the simplicity of the naive Bayes paradigm, several improvements can be found in the literature. Improvement in preprocessing, parameter estimation, feature selection and Bayesian approaches has been presented. Details can be found in Larrañaga (2003).

8.3 Seminaive Bayes

In spite of its simplicity, the naive Bayes algorithm cannot notice dependencies between predictive variables. Furthermore, there are some well-known databases where naive Bayes obtains poor performance Pazzani, 1997, perhaps because of its inability to discover any relationships between variables. A possible explanation of this matter is that the assumption of conditional independence is violated. To tackle this issue, the *seminaive Bayes* model was proposed. Although Kononenko (1991) introduces the *seminaive Bayes* classification model, the paper usually referred to is Pazzani (1997). In the work presented in Kononenko (1991), the seminaive Bayes classifier tries to avoid the assumptions of the classical naive Bayes taking into account new

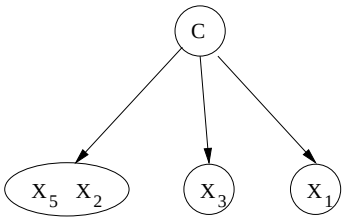


Fig. 8.2. Structure of a seminaive Bayes model.

FSSJ
- Step 1: Let the variable list, S, be empty
- Step 2: Repeat until non-improvement is reached
  - 2.1: Select the 'best' option
    - (a): Consider each predictive variable not in S a new variable conditionally independent of the class variable
    - (b): Consider joining each predictive variable not in S to each predictive variable in S
  - 2.2: Add the selected variable to S

Fig. 8.3. Pseudocode of the FSSJ algorithm Pazzani, 1997.

variables. These new variables consist of the values of the cartesian product of domain variables which overcome a condition. The condition is related to the independence concept and the reliability on the conditional probability estimations. It must be noted that, in the special cases in which all values meet the condition, the new variable is a cartesian product.

Pazzani (1996) presents the constructive induction concept. Naive Bayes and K-NN classifiers are induced by means of cartesian products of problem variables and a backward greedy wrapper search algorithm. This work is extended in the well-known Pazzani (1997) paper. This paper proposes a greedy wrapper approach to build a naive Bayes model where the irrelevant variables are removed and the correlated variables are joined in a cartesian product. Figure 8.2 displays a seminaive Bayes classifier structure. Two different algorithms are proposed, *Forward Sequential Selection and Joining* (FSSJ) –see Figure 8.3– and *Backward Sequential Elimination and Joining* (BSEJ) –see Figure 8.4–. The guide of the greedy search algorithms is the estimated accuracy (by a 10-fold cross validation or a leave-one-out validation) of the evaluated solution.

Looking at Figure 8.3, it can be seen that the FSSJ algorithm is a forward wrapper greedy algorithm with estimated accuracy as the search process guide. It starts with an empty set of variables and labels all the examples with the most common class value. Thus, until non-improvement is reached, the method selects the most accuratest option at each step: either to include a

BSEJ
Step 1: Let S be the variable list with all the predictive variables
Step 2: Repeat until non-improvement is reached
2.1: Select the 'best' option
(a): Consider replacing each pair of variables in S with a new variable: the cartesian product of the two variables
(b): Consider deleting each predictive variable in S
2.2: Delete (or replace) the selected variable(s) from S

Fig. 8.4. Pseudocode of the BSEJ algorithm Pazzani, 1997.

new variable or to join an existing variable with a new variable. The joining is performed by means of a cartesian product. The BSEJ algorithm is the backward version of FSSJ.

The *Large Bayesian* (LB) algorithm Meretakis and Wüthrich, 1999, based on Lewis (1959), proposes an approximation of the joint probability $p(c|\boldsymbol{x})$. The approach searches subsets of the variable values for each class value c. Since different subsets of variables are taken into account at different examples, LB could be regarded as a lazy classifier. The experimental results show that LB is comparable to naive Bayes and TAN.

8.4 Tree augmented naive Bayes

The seminaive Bayes model builds supernodes to represent the dependencies between the domain variables. Nevertheless, not all the dependencies between variables can be reached. In fact, in a supernode with several variables, seminaive Bayes considers that all the variables are related. Hence, it could include dependencies which do no exist in the domain. The *tree augmented naive Bayes* (TAN) classifier takes into account relationships between the predictive variables. A naive Bayes structure is extended with a tree structure among the predictive variables. Figure 8.5 shows a TAN model.

The adaptation of the Chow-Liu Chow and Liu, 1968 algorithm to build a TAN classifier is proposed by Friedman et al. (1997). The *construct-TAN* – see Figure 8.6– algorithm measures *conditional mutual information* instead of mutual information. The conditional mutual information of X and Y variables given the class C is defined as:

$$I(X,Y|C) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{c=1}^{w} p(x_i,y_j,c_r)\log\frac{p(x_i,y_j|c_r)}{p(x_i|c_r)p(y_j|c_r)}$$

The construct-TAN starts measuring the conditional mutual information for each pair of variables. Then, a complete undirected graph, whose nodes

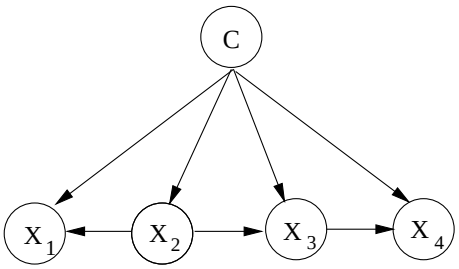


Fig. 8.5. Structure of a tree augmented naive Bayes model.

TAN

Step 1: Compute $I(X_i, X_j \mid C)$ between each pair of variables with $i < j$, $i, j = 1, \ldots, n$

Step 2: Build a complete undirected graph in which the vertexes are the predictive variables $X_1, X_2, \ldots, X_n$. Set the weight of an edge connecting X_i to X_j by $I(X_i, X_j \mid C)$

Step 3: Follow the Kruskal algorithm to build a maximum weighted spanning tree from the previous complete undirected graph

Step 4: Transform the resulting undirected tree into a directed tree by choosing a variable as the root and setting the direction of the edges

Step 5: Construct a TAN model by adding a C vertex and adding an arc from C to each X_i predictive variable

---

Fig. 8.6. Pseudocode for the adaptation to TAN classifier of the Chow-Liu algorithm Friedman et al., 1997.

are the predictive variables, is built. For each edge, its weight is set as the conditional mutual information of the two variables joined by the edge. Thus, the Kruskal algorithm uses the $n(n-1)/2$ weights to induce a maximum weighted spanning tree as follows:

1. Set the two edges with the major weight to the tree.
2. Look at the next edge with the major weight. If the addition of this edge makes a cycle, ignore it and look at the next one. If not set it to the tree.
3. Repeat Step 2 until selecting $n-1$ edges.

The construct-TAN has the same theoretical properties as the Chow-Liu algorithm, that is, the construct-TAN is asymptotically correct if the data have been generated by means of a TAN structure. This means that if the data size is large enough, the construct-TAN is able to induce the original TAN structure.

A discretisation approach by means of a dual representation in a TAN model is proposed by Friedman et al. (1998), that is, the continuous variables

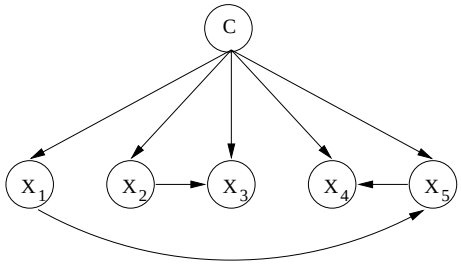


Fig. 8.7. Structure of a FAN model Lucas, 2004.

and the corresponding discrete variables are represented in the same probabilistic model. Nevertheless, this approach could be used in any structure. Three models are empirically compared: the discrete TAN, the continuous TAN and the dual TAN. The dual TAN is built using the discrete TAN and the continuous TAN and adding the dependencies between the continuous and discrete variables. The experimental results of the dual TAN are better than the results of the discrete and continuous models but without statistically significant differences.

A wrapper greedy approach to induce a TAN structure is presented in Keogh and Pazzani (1999). It starts with a naive Bayes structure and, at each step, adds an arc between two predictive variables bearing in mind the TAN condition. In the experimental results, the method proposed outperforms construct-TAN. The *SuperParent* (SP) heuristic is proposed to reduce the computational cost. The idea is to first find a good parent and then find the best child of that parent. In order to tackle the difficulties of the greedy search, Pernkopf and O'Leary (2003) proposes the use of the floating methods –presented in Section 5.2– to learn a TAN model.

The main restriction of the TAN model is the existence of exactly one parent for each variable of the problem domain except the root. To overcome this restriction, Lucas (2004) presents the *Forest Augmented Network* (FAN) algorithm ,that could be seen as a variation of TAN. A $k < n - 1$ number, with n predictive variables, must be fixed in order to cost the minimum forest –each edge is weighted with $I(X_i, X_j|C)$– with exactly k edges. Figure 8.7 shows a FAN structure.

The *Tree-Augmented Naive Credal* classifier (TANC algorithm) Fagiuoli and Zaffalon, 2000 builds credal Bayesian networks with a TAN structure. Although it is promising approach, no experiments have been carried out yet.

Another improvement of the TAN model is parameter transformation. For instance, Wettig et al. (2002) proposes a probabilistic graphical model based on classifier induction with parameter transformation. This parameter transformation finds the parameters that maximise the conditional likelihood of the naive Bayes and TAN models and any other Bayesian network model.

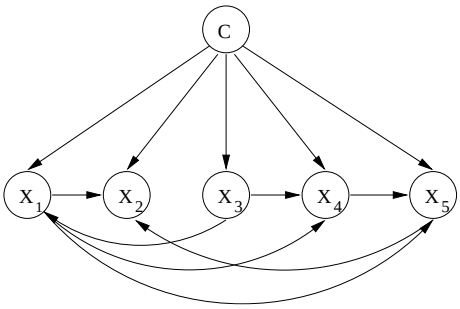


Fig. 8.8. Structure of a 3 dependence Bayesian classifier.

Parameter transformation makes the surface under the conditional likelihood convex-like. Then, any local search algorithm could reach the optimum.

The *Archs in Correct Order* (ACO) Roure, 2002 method to induce TAN structures starts with a TAN structure which is checked when new examples make the structure non-valid. The structure check is performed when the branches made by the new examples are not in the same order as the one provided by the current structure.

A Bayesian approach to TAN induction Cerquides, 1999 averages out the TAN model in a local way due to the difficulty of the global way. Cerquides and López de Mántaras (2003) introduces the decomposable distribution over TAN structures. This fact allows the expression resulting from Bayesian model averaging to be integrated into a closed form.

8.5 k dependence Bayesian classifier

The tree augmented naive Bayes classification model is limited by the number of parents of the predictive variables. A predictive variable can have a maximum of two parents: the class and another predictive variable. The k dependence Bayesian classifier (kDB) Sahami, 1996 tries to avoid this restriction by allowing a predictive variable to have up to k parents aside from the class. The definition of this model allows there to be structures as simple as naive Bayes and as complex as a complete Bayesian network. The k dependence classifier concept is defined where the parameter k must be fixed. Then, a naive Bayes structure is extended in such a way that each predictive variable could have k parents and the class variable. Therefore, naive Bayes could be seen as a $k = 0$ dependence Bayesian classifier, TAN is a $k = 1$ dependence Bayesian classifier and the complete Bayesian classifier is a $k = (n - 1)$ dependence Bayesian classifier. Figure 8.8 shows a kDB structure with $k = 3$ and Figure 8.9 presents the pseudocode of the kDB induction algorithm.

The construct-TAN Friedman et al., 1997 is generalised in the kDB algorithm. In this way, each variable can have k predictive variables as parents

kDB
Step 1: For each predictive variable X_i, compute $I(X_i, C)$
Step 2: Compute $I(X_i, X_j \mid C)$ between each pair of variables with $i < j$, $i, j = 1, \ldots, n$
Step 3: Let the used variable list, S, be empty
Step 4: Let the k dependence Bayesian classifier being constructed, kDB, begin with a single class node, C
Step 5: Repeat until S includes all predictive variables
5.1: Select the variable X_{max} which is not in S and has the largest $I(X_{max}, C)$
5.2: Add a node to kDB representing X_{max}
5.3: Add an arc from C to X_{max} in kDB
5.4: Add $m = \min(|S|, k)$ arcs from m different variables X_j in S with the highest value for $I(X_{max}, X_j \mid C)$
5.5: Add X_{max} to S

Fig. 8.9. Pseudocode for the kDB algorithm Sahami, 1996.

apart from the class variable. However, this number is fixed a priori by hand, and it cannot be optimum. Moreover, all the variables (except the first k variables introduced) have the same number of parents, k, without counting the class. This fact is reported in the original paper by Sahami (1996) and an improvement is proposed. A mutual information threshold is fixed in order to not include arcs between the less correlated variables. Then, k is a upper bound of the number of parents.

A similar algorithm called the *Augmented Naive Bayes* model is proposed by Zhang and Ling (2001). The only restriction is the non-allowance of cycles in the model. Then, a naive Bayes structure is extended and each predictive variable can have a different number of parents. It is shown that this kind of model can represent any Bayesian classifier.

9

New methods to learn supervised Bayesian classification models

Bayesian classifiers are promising models to perform classification tasks. However, problem domains appearing in the last years are practically intractable with the classical Bayesian classifiers presented in Chapter 8. Thus, new methods for Bayesian classifier induction based on filter and wrapper ideas of the feature subset selection are proposed in this chapter.

The chapter is organised as follows. Section 9.1 briefly introduces both the filter and wrapper approaches to Bayesian classifier induction. In Section 9.2, the novel filter approach to Bayesian classification models is described in detail. In the same way, the wrapper approach is explained in Section 9.3. Finally, experimental results of the methods proposed over synthetic and UCI datasets are presented in Section 9.4.

9.1 Introduction

The classical approaches to Bayesian classifiers –see Chapter 8– induction include all the variables of the problem domain. However, domain variables could be redundant or irrelevant and hence decrease the performance of the classifier. The performance of the naive Bayes model (and, generally, the Bayesian classifiers) does not decrease when irrelevant variables are included in the model. In order to explain this issue, suppose that X_i is an irrelevant variable added to naive Bayes model. Then:

$$p(c|x_1,\ldots,x_{(i-1)},x_i,x_{(i+1)},\ldots,x_n) \propto p(c)p(x_i|c)\prod_{l=1}^{i-1}p(x_l|c)\prod_{l=i+1}^{n}p(x_l|c)$$

where $p(x_i|c) = p(x_i|\bar{c})$. This way:

$$p(c|x_1,\ldots,x_{(i-1)},x_i,x_{(i+1)},\ldots,x_n) \propto p(c|x_1,\ldots,x_{(i-1)},x_{(i+1)},\ldots,x_n).$$

Nevertheless, the redundant variables have a great influence on the accuracy of the naive Bayes model (and generally the Bayesian classifiers). Suppose

that X_i is a redundant variable included in the naive Bayes model in such a way that $X_i = X_{(i-1)}$. Then:

$$p(c|x_1,\ldots,x_j,\ldots,x_{(i-1)},x_i,x_{(i+1)},\ldots,x_n) \propto p(c)p(x_{(i-1)}|c)p(x_i|c)\prod_{l=1}^{i-2}p(x_l|c)\prod_{l=i+1}^{n}p(x_l|c)$$

where $p(x_{i-1}|c) = p(x_i|c)$. Hence, the following formulation is obtained:

$$p(c|x_1,\ldots,x_j,\ldots,x_{(i-1)},x_i,x_{(i+1)},\ldots,x_n) \propto p(c)(p(x_i|c))^2\prod_{l=1}^{i-2}p(x_l|c)\prod_{l=i+1}^{n}p(x_l|c).$$

In this way, the performance of naive Bayes suffers when redundant variables are taken into account in the classifier.

Moreover, new domains such as medical domains have a relatively small number of examples in relation to the number of variables due to the difficulty of the collection process. Therefore, the classifiers that take into account complex relationships between the domain variables are unreliable and unstable.

The *filter* and *wrapper* approaches to feature selection could be adapted to Bayesian classifier induction to avoid these difficulties. As in the feature selection task, the building of a Bayesian classifier involves a search process in the space of classifier structures in order to maximise the accuracy of the final classifier(indirectly by means of a measuring function in the case of a filter approach or directly by means of accuracy in the case of a wrapper approach). These approaches to Bayesian classifier induction are inspired by filter and wrapper approaches to feature subset selection where the same ideas are used to search for the 'best' subset of features.

9.1.1 Filter approach

When the filter approach is considered, a function independent of the characteristics of the specific classifier must be set. Functions like likelihood, conditional likelihood, entropy or mutual information are commonly used to indirectly measure the classifier's power to distinguish between the states of the domain class.

Likelihood is a scoring function frequently used to learn Bayesian network structures in a score+search process, which could be applied to Bayesian classifier induction. Due to computational reasons, likelihood is usually calculated applying a logarithm. Hence, it is defined as:

$$LL(B|D) = \sum_{l=1}^{N}\log p(c^l,\boldsymbol{x}^l|\boldsymbol{\theta})$$

where B is the Bayesian classifier and D the given dataset.

However, $LL(B|D)$ is usually considered with additional penalisation terms to control its tendency to favour more complex structures. Penalisation terms are included and new scoring metrics like Akaike's Information Criterion (AIC) Akaike, 1974 and Bayesian Information Criterion Schwarz, 1978 are proposed.

The *minimum description length* Rissanen, 1978 is not a penalised score, but it is a function of the $LL(B|D)$ measure defined as:

$$MDL(B|D) = \frac{1}{2}|B| \log N - LL(B|D)$$

where $|B|$ is the number of parameters of the network. The $MDL(B|D)$ function has been used with classification purposes Friedman and Goldszmidt, 1996; Kontkanen et al., 2000.

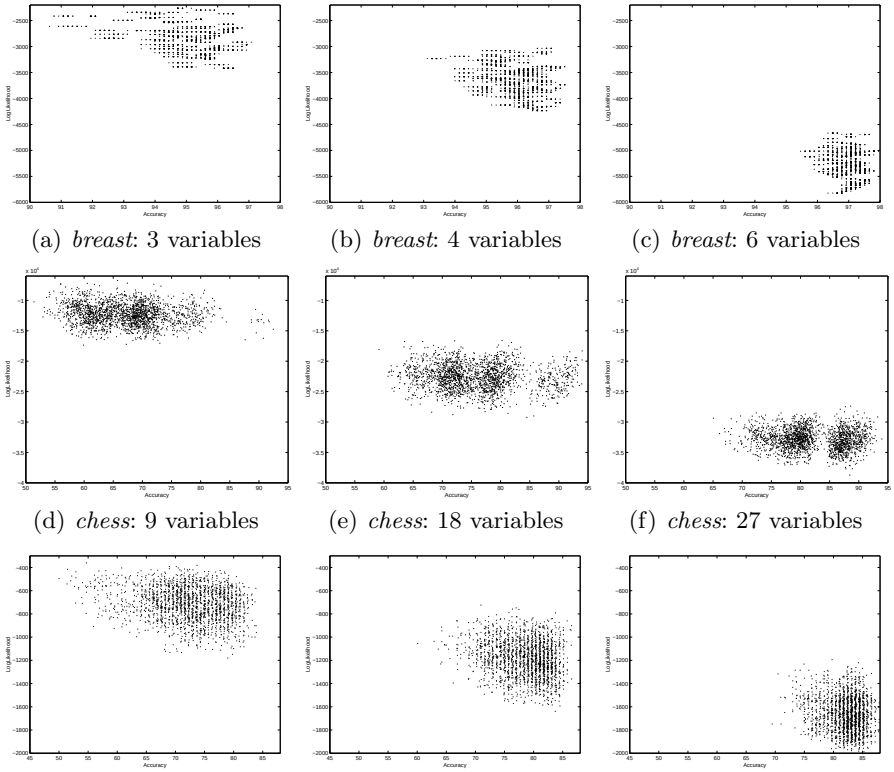
In a sense, it could be assumed that a classifier with higher $LL(B|D)$ attains better results than others with lower $LL(B|D)$. Nevertheless, in order to prove this assumption and study the relationship between the $LL(B|D)$ function and the accuracy estimation, an empirical experimentation over a set of databases of the UCI Repository Blake and Merz, 1998 is carried out. First, in order to test the existence of a relationship between $LL(B|D)$ and accuracy, 2000 selective naive Bayes models are randomly generated fixing beforehand the number of selected variables (with the same number of states), since the $LL(B|D)$ takes into account the number of variables and their states. Then, the corresponding accuracy and $LL(B|D)$ measures are calculated. The plots appear in Figures 9.1 and 9.2, where the x axis represents accuracy and the y axis the log-likelihood.

Except with the *splice* database, Figures 9.1 and 9.2 show an almost random behaviour between the $LL(B|D)$ function and the accuracy estimation. This seems to be caused by $LL(B|D)$'s tendency to favour complex structures. Then, a penalised likelihood score should correct this tendency. Following the former experimentation scheme, the BIC score behaviour with respect to accuracy is tested. Figures 9.3 and 9.4 show the results of this experimentation.

It must be noted that, as in the previous experimentation, the relationship between the BIC score and accuracy appears only in the *splice* dataset. There does not seem to be any related behaviour in the other datasets.

As the BIC score penalisation takes into account the number of variables in the network (classifier) and the number of states of each variable, the same experimentation is carried out without fixing the number of variables included in the model. These experimental results are shown in Figure 9.5. It can be seen that there is no relation between the BIC score and accuracy except in the *breast* dataset. In this case, the graph surprisingly shows an unexpected behaviour, as the BIC increases when accuracy decreases.

In spite of the assumption that the $LL(B|D)$ scoring function increases when accuracy increases, the empirical experimentation shown in Figures 9.1, 9.2, 9.3, 9.4 and 9.5 does not support this issue. In fact, many datasets do not



(a) *breast*: 3 variables (b) *breast*: 4 variables (c) *breast*: 6 variables

(d) *chess*: 9 variables (e) *chess*: 18 variables (f) *chess*: 27 variables

(g) *lymphography*: 5 variables (h) *lymphography*: 9 variables (i) *lymphography*: 13 variables

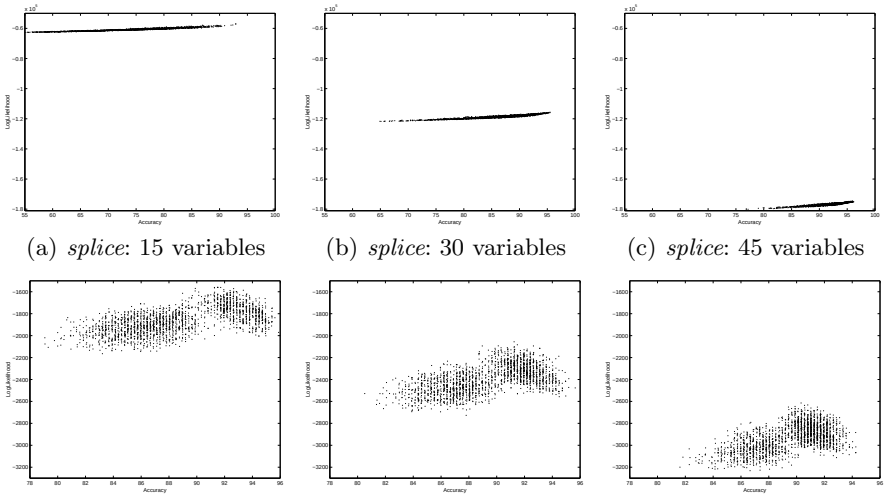
Fig. 9.1. Plot of the relation between accuracy and $LL(B|D)$ for the *breast*, *chess* and *lymphography* databases.

show a clear relationship between the $LL(B|D)$ (or the penalised $LL(B|D)$) function and the accuracy estimation. Moreover, in the empirical experimentation with the BIC, an unclear relationship can be observed in the *breast* dataset: as the BIC score increases, the estimated accuracy decreases. Therefore, the use of the $LL(B|D)$ (or the penalised $LL(B|D)$) measure to guide the search of a classification model is not always advisable.

The *conditional log-likelihood* is a score used in classic Bayesian network learning defined as:

$$CLL(B|D) = \sum_{l=1}^{N} \log p(c^l|\boldsymbol{x}^l, \boldsymbol{\theta})$$

When using $CLL(B|D)$ for classification, maximising the $CLL(B|D)$ is equivalent to maximising the ability of class prediction for each instance Friedman et al., 1997. However, the maximum $CLL(B|D)$ cannot be efficiently com-



(a) *splice*: 15 variables (b) *splice*: 30 variables (c) *splice*: 45 variables

(d) *vote*: 6 variables (e) *vote*: 8 variables (f) *vote*: 10 variables

Fig. 9.2. Plot of the relation between accuracy and $LL(B|D)$ for the *splice* and *vote* databases.

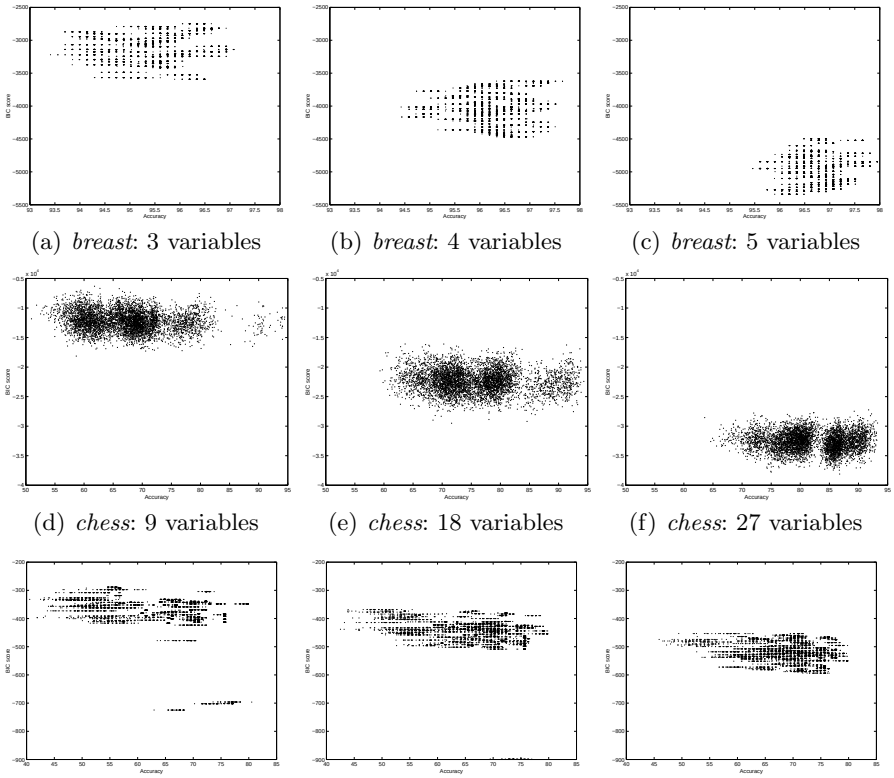
puted and Friedman et al. (1997) suggests the use of heuristics. This idea is collected in some articles. For instance, Greiner and Zhou (2002) proposes a discriminative Bayesian classifier inductor which looks for the parameters that maximise $CLL(B|D)$. More recently, Grossman and Domingos (2004) applies the hill-climbing algorithm guided by $CLL(B|D)$ to learn a Bayesian network with classification purposes.

The *mutual information* of two variables is a filter function defined as:

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

It could be interpreted as the reduction in uncertainty about one variable due to the knowledge of another variable. The mutual information between a predictive variable X and the class C, $I(X,C)$, that is, the reduction in uncertainty about C due to the knowledge of the predictive variable X, is largely applied for classification purposes. Bayesian classification inducers like construct-TAN Friedman et al., 1997 and kDB Sahami, 1996 require mutual information to build the classifier proposed.

It has not systematically been studied whether there is any relationship between the mutual information and the accuracy of a classifier. In order to prove the existence of any related behaviour, an empirical experimentation, similar to the former experimentations, is carried out. Due to the mutual information formula, which takes into account the number of variables and the number of states of each variables, the number of selected variables with



(a) *breast*: 3 variables (b) *breast*: 4 variables (c) *breast*: 5 variables

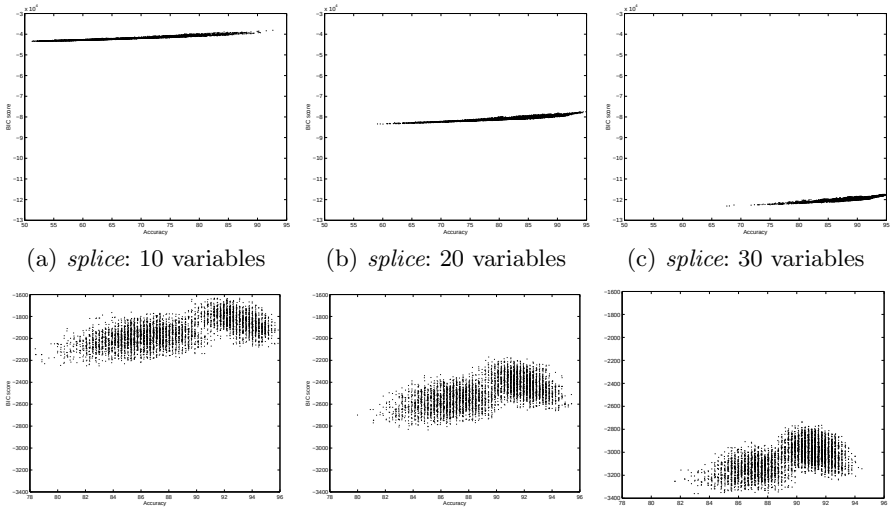
(d) *chess*: 9 variables (e) *chess*: 18 variables (f) *chess*: 27 variables

(g) *lymphography*: 3 variables (h) *lymphography*: 4 variables (i) *lymphography*: 5 variables

Fig. 9.3. Plot of the relation between accuracy and the BIC scoring measure for the *breast*, *chess* and *lymphography* databases.

the same number of states is fixed beforehand. Thus, for each dataset, 2000 selective naive Bayes are randomly generated, and their accuracy and mutual information, calculated. Figures 9.6 and 9.7 show the results. It must be noted that a kind of relationship could clearly be observed, except in the *breast* and *vote* databases, where the relationships are not so clear.

This empirical experimentation turns the mutual information into a feasible filter measure to perform a filter approach to Bayesian classifier induction. However, due to the measure increment if the number of values increases a threshold is desirable. It is known Pardo, 1997 that $2N\,I(X_i, C)$ (where N is the number of input instances) asymptotically follows a $\chi^2_{(r_i-1)(r_0-1)}$ distribution where r_i is the number of values of the X_i variable and r_0 is the number of values of the class. By means of this outcome, a threshold could be objectively set.



(a) *splice*: 10 variables (b) *splice*: 20 variables (c) *splice*: 30 variables

(d) *vote*: 6 variables (e) *vote*: 8 variables (f) *vote*: 10 variables

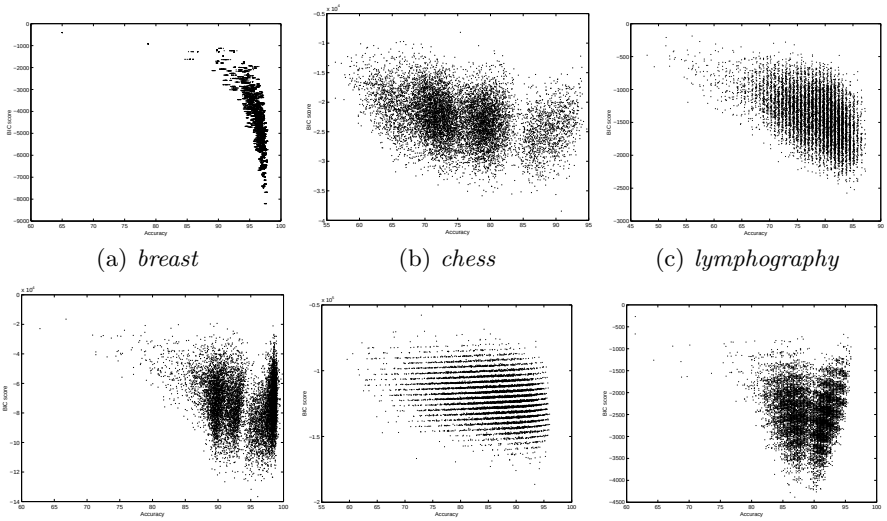
Fig. 9.4. Plot of the relation between accuracy and the BIC scoring measure for the *splice* and *vote* databases.

9.1.2 Wrapper approach

When a wrapper approach is considered in the construction of the classifier, the estimated accuracy of the classifier over instances seen is the function which guides the search process for classifier induction –see Chapter 8 for Bayesian classifier wrapper inducers in the literature–. However, not only is accuracy estimation a wrapper measure. The area under the ROC curve Hanley and McNeil, 1982 and the Brier score Brier, 1950 are considered wrapper measures due to the participation of the classification model when computing these measures.

Receiver Operating Characteristics (ROC) Egan, 1975 curves are a useful technique to organise classifiers and to visualise their performance. ROC curves are commonly used in medical decision making Bergus, 1993; Ward, 1986, and, for the past years, have been increasingly adopted by the machine learning field Adams and Hand, 1999; Aliferis et al., 2003; Bradley, 1997; Spackman, 1989. A ROC graph plots relative trade-offs between benefits (sensibility) and cost (1− specificity). Figure 9.8 shows an example of the ROC curve. The more distant the curve from the diagonal, the better the classifier.

A Receiver Operating Characteristic (ROC) analysis is useful to analyse the cost–benefit ratio of diagnostic decision making. Furthermore, the ROC curve is proven to be a better evaluation measure than accuracy in problem domains with unbalanced class distribution Fawcett, 2004. The area under



(a) *breast* (b) *chess* (c) *lymphography*

(d) *mushroom* (e) *splice* (f) *vote*

Fig. 9.5. Plot of the relation between accuracy and the BIC scoring measure without a beforehand fixed number of variables.

the ROC curve could be the scoring function to guide a search process in a wrapper way to induce a classifier.

A Bayesian classifier is a probabilistic network. The posterior distribution of the classifier is used to label new instances and the most probable class reported in order to calculate the accuracy. However, there is another way to measure classifier outcome: the posterior distribution of the different class states. From this point of view, a Bayesian classifier becomes a probabilistic forecaster due to the predictions made regarding the feasible outcome. The quality of a probabilistic forecaster is often expressed in terms of *calibration*. The calibration score describes how close the estimates of the model are to the true underlying probabilities Vinterbo, 1999. As the true probability is unknown, a calibration score cannot be calculated directly.

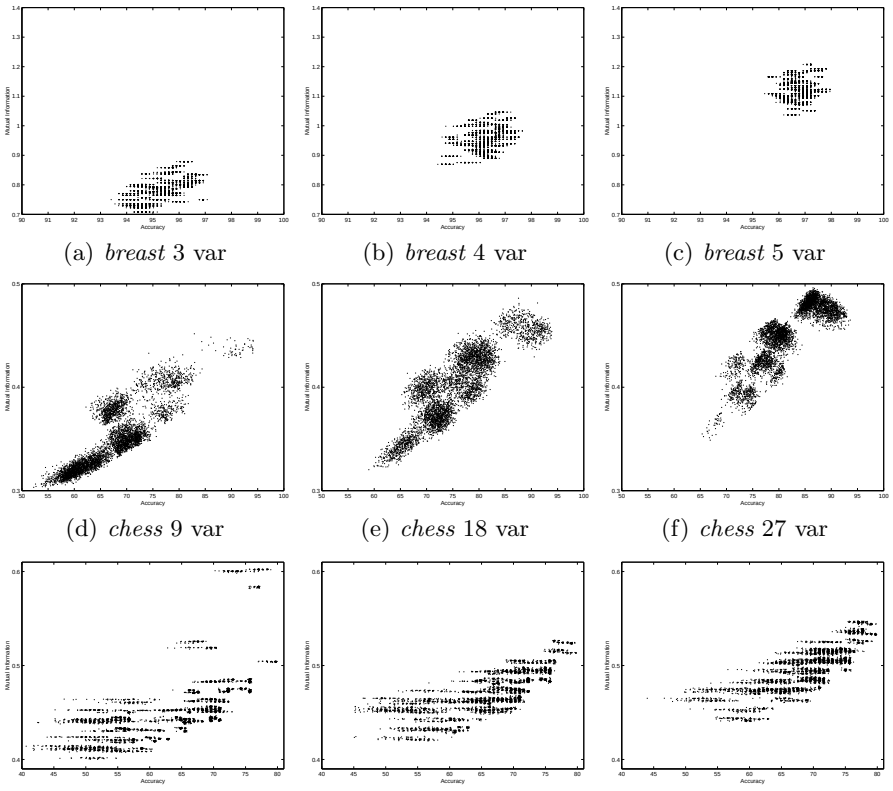
The Brier score Brier, 1950 is a well-known calibration measure applied especially to the weather forecast field Murphy, 1972; Panofsky and Brier, 1968. Following the formulation in van der Gaag and Renooij (2001), it is defined as:

$$bs(D) = \frac{1}{N}\sum_{l=1}^{N}\sum_{k=1}^{r_0}(p(C = c_k|\boldsymbol{X} = \boldsymbol{x}^l) - \delta_{l,k})^2$$

where

$$\delta_{l,k} = \begin{cases} 1 & c^l = c_k \\ 0 & \text{otherwise} \end{cases}$$

The Brier score denotes whether the estimated value c_k of the class equals the real value c^l of the class in instance $\boldsymbol{x}^l$. It must be noted that the lower the



(a) *breast* 3 var (b) *breast* 4 var (c) *breast* 5 var

(d) *chess* 9 var (e) *chess* 18 var (f) *chess* 27 var

(g) *lymphography* 3 var (h) *lymphography* 4 var (i) *lymphography* 5 var

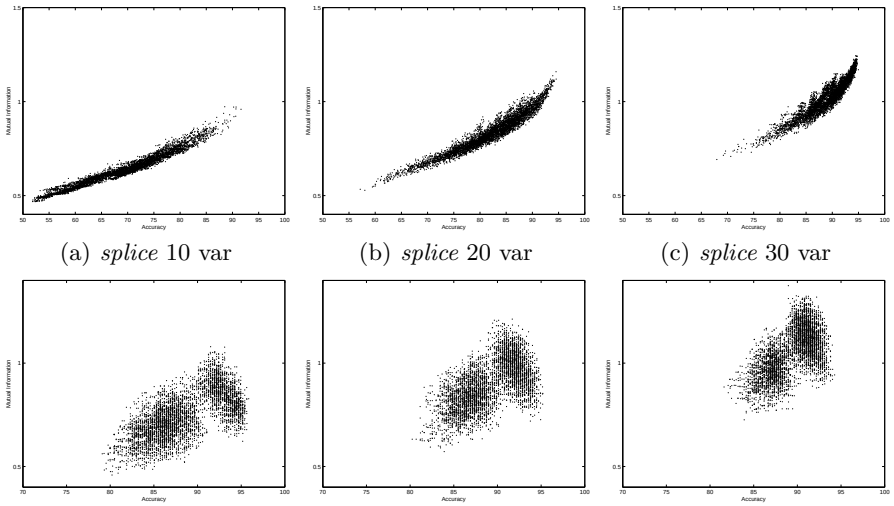
Fig. 9.6. Plot of the relation between accuracy and mutual information for the *breast*, *chess* and *lymphography* databases.

score the better the classifier. The induction of a classifier could be tackled in a wrapper approach using the Brier score. In this case, the best solution is the solution with the lowest Brier score.

In this work, the estimated accuracy is the guide of the search. Nevertheless in the medical applications performed, the ROC curve and the Brier score are taken into account and displayed for the sake of clarity.

9.2 Filter approach to Bayesian classifier induction

Mutual information is a commonly used measure for feature selection Lewis, 1992; Blum and Langley, 1997; Zaffalon and Hutter, 2002 and Bayesian classifier induction Friedman et al., 1997; Sahami, 1996; Duda et al., 2001. Therefore, it becomes an attractive measure to perform a filter approach to Bayesian classifiers with implicit feature reduction. However, due to the tendency to



(a) *splice* 10 var (b) *splice* 20 var (c) *splice* 30 var

(d) *vote* 6 var (e) *vote* 8 var (f) *vote* 10 var

Fig. 9.7. Plot of the relation between accuracy and mutual information for the *splice* and *vote* databases.

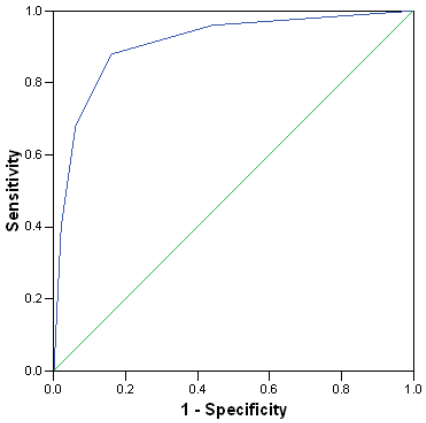


Fig. 9.8. An example of ROC curve.

favour complex structures when all the variables are included in the model, a sophisticated approach has to be developed in order to improve the estimated accuracy.

It is known Pardo, 1997 that $2NI(X_i, C)$ asymptotically follows a χ^2 probability distribution with $(r_i - 1)(r_0 - 1)$ degrees of freedom. Thus, given the mutual information of a predictive variable and the class value, a $\chi^2_{(r_i-1)(r_0-1);1-\alpha}$ test could be performed to check the former property.

Selective NB_{fs}
  Step 1: Let the variable list, S, be empty
  Step 2: Repeat until all the variables have been seen
    2.1: If $2N\,I(X_i, C)$ surpasses the $\chi^2_{(r_i-1)(r_0-1);1-\alpha}$ value then add X_i to S

Fig. 9.9. Pseudocode for the filter approach to selective naive Bayes.

By means of the novel adaptation to Bayesian classifiers induction of this outcome, rejection of the irrelevant variables in the final classifier is expected. Moreover, the use of this statistical filter measure provides the learning of Bayesian classifier structures and the feature reduction process with robustness and reliability. Therefore, this theoretical result is adapted to the induction of the selective naive Bayes, seminaive Bayes, TAN and kDB Bayesian classification models.

In the novel proposed filter approach to selective naive Bayes, only the subset of variables whose $2NI(X_i, C)$ surpasses the $\chi^2_{(r_i-1)(r_0-1);1-\alpha}$ value is selected. Then, a naive Bayes structure is learnt with this subset of variables. In this way, it is expected that the irrelevant variables disappear in the final naive Bayes classifier. Figure 9.9 shows the pseudocode of this approach.

Once the filter selective naive Bayes is explained, it must be noted that, as explained before, it cannot detect the dependencies between the domain variables. The theoretical result related to mutual information is therefore adapted to obtain a filter approach to seminaive Bayes. This filter approach follows the general scheme proposed by Pazzani (1997) –see Section 8.3–. Instead of a wrapper estimation of the accuracy, the 'best' option is the choice with the highest percentile of the χ^2 test between:

- Add a new variable X_{new} conditionally independent of the class variable whose $2NI(X_{new}, C)$ surpasses the $\chi^2_{(r_{new}-1)(r_0-1);1-\alpha}$ value.
- Replace a variable X_i by joining X_i to a new variable, X_{new}, resulting in $X_{join} = (X_i, X_{new})$, so that $2NI((X_i, X_{new}), C)$ surpasses the $\chi^2_{((r_i r_{new})-1)(r_0-1);1-\alpha}$ value.

It must be remarked that the degrees of freedom change when checking a cartesian product. Due to the fact that a supernode is composed of a subset of predictive variables, the number of states of that supernode increases with respect to a single variable. Thus, the degrees of freedom should increase. The pseudocode of the filter approach to seminaive Bayes can be seen in Figure 9.10. In the special case in which no cartesian passes the test, this filter approach to seminaive Bayes becomes in the selective naive Bayes classifier.

As explained in Section 8.4, the seminaive Bayes model cannot reach all dependencies between the variables included in the model. In this filter approach, the same issue appears. Hence, the TAN can provide flexibility to

Semi NB_{fs}
Step 1: Let the variable list, S, be empty
Step 2: Repeat until no variables whose $2N\,I(X_i, C)$ surpasses the $\chi^2_{(r_i-1)(r_0-1);1-\alpha}$ value
2.1: Select the option with the highest percentile
(a): Consider each predictive variable X_{max} not in S so that $2N\,I(X_{max}, C)$ surpasses the $\chi^2_{(r_{max}-1)(r_0-1);1-\alpha}$ test as a new variable conditionally independent of the current classifier
(b): Consider joining each predictive variable X_{new} not in S to each predictive variable X_i in S so that $2N\,I((X_i, X_{new}), C)$ surpasses the $\chi^2_{((r_i r_{new})-1)(r_0-1);1-\alpha}$ value
2.2: Add the selected variable to S

Fig. 9.10. Pseudocode for the filter approach to seminaive Bayes.

represent the relationships of the domain. Nevertheless, the definition of the TAN model given in Friedman et al. (1997) does not allow forests in the model. This means that a tree structure connecting all the problem variables must be made. This fact makes the TAN structure more restrictive than the forest to represent the relationships of the domain. Therefore, sometimes the TAN structure is not the most accurate model and a forest classifier is convenient.

Following the novel filter scheme proposed, the construct-TAN algorithm proposed by Friedman et al. (1997) –see Section 8.4– is adapted to select a subset of variables and a subset of all the possible arcs between the features. Hence, in order to adapt the filter idea to the construct-TAN method, the mutual information of each predictive variable and the class $I(X_i, C)$, and the conditional mutual information of each pair of domain variables given the class $I(X_i, X_j|C)$, are taken into account. However, the main problem is the non-existence of a statistic with a known distribution to fix the significance of $2NI(X_i, X_j|C)$ where X_i and X_j are the variables related to the arc. To solve this issue, many solutions could be proposed, but in this work it is required that $2N_cI(X_i, X_j|C = c)$ pass the $\chi^2_{(r_i-1)(r_j-1);1-\alpha}$ test to at least a single value c of the class variable. It must be remarked that, then, N_c is the number of cases where $C = c$. In this way, the inclusion of irrelevant arcs between relevant variables is not expected.

The construct-TAN algorithm is performed over the subset of variables whose $2NI(X_i, C)$ surpasses the $\chi^2_{(r_i-1)(r_0-1);1-\alpha}$ value. Then, the Chow-Liu Chow and Liu, 1968 algorithm is carried out, but only the arcs whose $2N_cI(X_i, X_j|C = c)$ surpasses the $\chi^2_{(r_i-1)(r_j-1);1-\alpha}$ value to at least a single value c are added to the undirected graph. It is feasible for more than one connected component to appear. Hence, the root is randomly selected in each connected component and the edges, directed. Finally, the arcs between the class and the selected variables are included in the model. In this way, a forest

TAN_{fs}

Step 1: Select the subset S of variables so that $2N\,I(X_i, C)$ surpasses the $\chi^2_{(r_i-1)(r_0-1);1-\alpha}$ value

Step 2: Compute $I(X_i, X_j \,|\, C)$ between each pair of variables in S with $i < j$, $i, j = 1, \ldots, |S|$

Step 3: Build a complete undirected graph in which the vertexes are the predictive variables in S. If $2N_c\,I(X_i, X_j|C = c)$ surpasses the $\chi^2_{(r_i-1)(r_j-1);1-\alpha}$ value to some value c, then set the weight of an edge connecting X_i to X_j by $I(X_i, X_j \,|\, C)$

Step 4: Follow the Kruskall algorithm to build a maximum weighted spanning tree from the previous undirected graph. If the graph is not connected, repeat this step for each connected component

Step 5: Transform the resulting undirected tree(s) to (a) directed tree(s) by choosing a variable as the root and setting the direction of the edges

Step 6: Construct a TAN model by adding a C vertex and adding an arc from C to each predictive variable in S

Fig. 9.11. Pseudocode for the filter approach to TAN.

structure can be reached. Figure 9.11 shows the pseudocode for the proposed filter approach to the TAN model.

The original proposal of the kDB classification model suffers the same problem as the TAN classifier. The model proposed by Sahami (1996) does not allow unconnected components among the predictive variables. Moreover, the k parameter, which is the number of parents of a domain variable, is fixed by the user. For the domain variables to have as many dependencies as in the real problem domain, some flexibility is required. Therefore, the novel presented filter approach can be adapted to allow different number of parents for each variable. Then, k becomes the upper bound of the number of parents of a predictive variable. Furthermore, unconnected components are permitted.

The proposed novel filter approach to a kDB classifier, like the TAN model, uses mutual information and conditional mutual information. The problem of the filter approach to TAN appears in the filter approach to kDB. It is solved in the same way. As the existence of a statistic with a known distribution to fix the significance of $2NI(X_i, X_j|C)$ is unknown, $2N_cI(X_i, X_j|C = c)$, where X_i and X_j are the variables involved in the edge and N_c is the number of cases where $C = c$, should surpass the $\chi^2_{(r_i-1)(r_j-1);1-\alpha}$ value to at least a single value c of the class variable.

Then, the kDB algorithm proposed by Sahami (1996) is performed over the subset of variables whose $2NI(X_i, C)$ surpasses the test. When an edge is included, the former property is checked, and only the edges whose $2N_cI(X_i, X_j|C = c)$ surpasses the value are finally added. At the last step, the structure is augmented with naive Bayes. Thus, unconnected components are allowed and k

kDB$_{fs}$
Step 1: Select the subset V of variables so that $2N\, I(X_i, C)$ surpasses the $\chi^2_{(r_i-1)(r_0-1);1-\alpha}$ value
Step 2: Compute $I(X_i, X_j|C)$ between each pair of variables in V with $i < j$, $i, j = 1, \ldots, |V|$
Step 3: Let the used variable list, S, be empty
Step 4: Let the k-dependence Bayesian classifier being constructed, kDB, begin with a single class node, C
Step 5: Repeat until S includes all predictive variables in V
  5.1: Select the variable X_{max} in V which is not in S and has the largest $I(X_{max}, C)$
  5.2: Add a node to kDB representing X_{max}
  5.3: Add an arc from C to X_{max} in kDB
  5.4: Add $m = \min(|S|, k)$ arcs from m different variables X_j in S with the highest value for $I(X_{max}, X_j|C)$ if $2N_c\, I(X_{max}, X_j|C = c)$ surpasses the $\chi^2_{(r_i-1)(r_j-1);1-\alpha}$ value to some value c
  5.5: Add X_{max} to S

Fig. 9.12. Pseudocode for the filter approach to kDB.

becomes the upper bound of the number of parents of a predictive variable. Figure 9.12 shows the pseudocode of this approach to kDB.

9.3 Wrapper approaches to Bayesian classifier induction

When the main goal is to maximise classifier accuracy, looking at the plots presented in Section 9.1.1, the filter scoring measures do not always guarantee the highest accuracy. Furthermore, the computationally increasing power of computers has allowed the development of wrapper approaches to Bayesian classifiers since the beginning of the 90s.

The wrapper approach to selective naive Bayes is proposed by Langley and Sage (1994) in order to improve asymptotic accuracy in domains with correlated variables. The removal of the correlated variables is attained. This proposal is a forward greedy algorithm that, starting with an empty set of variables, induces a naive Bayes structure with a subset of domain variables. At each step the variable with the highest accuracy increase is added until non-improvement is reached. Figure 9.13 shows the pseudocode for the wrapper approach to selective naive Bayes.

The analogous backward version could be developed. In this case, the initial starting point is a naive Bayes structure with all the problem variables. At each step, the variable with the lowest accuracy increase is deleted, until non-improvement is attained.

Selective NB_{ws}
  Step 1: Let the variable list, S, be empty
  Step 2: Repeat until non-improvement is reached
    2.1: Select the most accurate predictive variable not in S
    2.2: Add the selected variable to S

Fig. 9.13. Pseudocode for the wrapper approach to selective naive Bayes Langley and Sage, 1994.

The wrapper greedy approach to the seminaive Bayes model was proposed by Pazzani (1997). The forward version is known as *FSSJ* and the backward one as *BSEJ*. See Section 8.3 for full details.

Wrapper approaches to induce a TAN classifier have been proposed in the literature Keogh and Pazzani, 1999; Pernkopf and O'Leary, 2003. However, these proposals include all the domain variables in the final model. They therefore allow the existence of a forest structure with all the variables included. Nevertheless, the model performance could decrease due to the inclusion of redundant variables. In order to overcome this issue, the proposed wrapper approach to the TAN model is defined in such a way that the classifier variables and the arcs between them are selected during the search process. Therefore, the novel proposed wrapper approach to the TAN model starts with an empty set of variables and, at the first step,the two most accurate variables are sequentially added. Then, at each step, the most accurate option between adding a new variable conditionally independent of class node, or including an arc between two variables previously added to the TAN classifier, is chosen. It must be remarked that, when considering arc addition, the condition of the tree structure must be checked. If arc addition makes the child node have more than one parent node, the arc is not taken into account. From this point of view, the wrapper approach to TAN could be regarded as the FSSJ algorithm where the joining of two variables is replaced by arc addition between the two nodes. Figure 9.14 shows the pseudocode for the wrapper greedy approach to TAN classifier.

The novel proposed wrapper approach to the kDB classifier is similar to the wrapper approach to the TAN model. As in the TAN model case, the original kDB model Sahami, 1996 does not allow unconnected components. However, the novel wrapper approach to kDB permits the presence of unconnected components. The wrapper greedy algorithm proposed is analogous to the wrapper approach to TAN but, in this case, when considering arc addition, the condition to obtain a kDB structure must be checked. The pseudocode of the method can be seen in Figure 9.15.

TAN_{ws}
Step 1: Let the variable list, S, be empty
Step 2: Sequentially select the two most accurate predictive variables, and add them to S
Step 3: Repeat until non-improvement is reached
3.1: Select the 'best' option
(a): Consider each predictive variable not in S a new variable conditionally independent of the class variable
(b): Consider each arc which does not invalidate the forest TAN condition between two predictive variables in S
3.2: Add the corresponding decision

Fig. 9.14. Pseudocode for the wrapper approach to TAN.

$k\text{DB}_{ws}$
Step 1: Let the variable list, S, be empty
Step 2: Sequentially select the two most accurate predictive variables, and add them to S
Step 3: Repeat until non-improvement is reached
3.1: Select the 'best' option
(a): Consider each predictive variable not in S a new variable conditionally independent of the class variable
(b): Consider each arc which does not invalidate the kDB condition between two predictive variables in S
3.2: Add the corresponding decision

Fig. 9.15. Pseudocode for the wrapper approach to kDB.

9.4 Experimental results

Once the Bayesian classifiers and the novel filter and wrapper approaches have been presented, an exhaustive experimental study is carried out. Both synthetic and real-world domains are used to analyse the quality of the novel approaches to Bayesian classification models. Although several experiments are performed, the parameters of the Bayesian classifiers are calculated using the Laplace correction Laplace, 1814 to their maximum likelihood parameter estimations. Each classifier is run over each domain in a 10-fold cross-validation framework. Moreover, in the filter approaches, the α parameter is fixed at $\alpha = 0.01$ and the k parameter of the kDB classifiers is fixed at $k = 3$.

It must be noted that the Bayesian classifiers presented in the previous section are implemented by Elvira software Elvira Consortium, 2002, and all the experimentation is performed using Elvira.

	synthetic-1											
	1000				5000				10000			
	acc.	±	*dev.*	*feat.* ± *dev.*	*acc.*	±	*dev.*	*feat.* ± *dev.*	*acc.*	±	*dev.*	*feat.* ± *dev.*
Naive Bayes	94.90	±	1.13	50.0 ± 0.00	99.86	±	0.12	50.0 ± 0.00	100.0	±	0.00	50.0 ± 0.00
TAN	90.00	±	3.49	50.0 ± 0.00	97.24	±	0.53	50.0 ± 0.00	98.03	±	0.42	50.0 ± 0.00
kDB	71.90	±	4.41	50.0 ± 0.00	93.30	±	1.56	50.0 ± 0.00	98.47	±	0.23	50.0 ± 0.00
Selective NB$_{fs}$	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00	4.0 ± 0.00
Semi NB$_{fs}$	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00	4.0 ± 0.00
TAN$_{fs}$	99.50	±	0.80†	4.0 ± 0.00	100.0	±	0.00†	4.0 ± 0.00	99.85	±	0.30	4.0 ± 0.00
kDB$_{fs}$	99.80	±	0.59†	4.0 ± 0.00	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00	4.0 ± 0.00
Selective NB$_{ws}$	100.0	±	0.00†	4.5 ± 0.53	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00	4.0 ± 0.00
Semi NB$_{ws}$	99.98	±	0.06†	5.4 ± 0.55	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00	4.0 ± 0.00
TAN$_{ws}$	100.0	±	0.00†	4.4 ± 0.51	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00	4.0 ± 0.00
kDB$_{ws}$	100.0	±	0.00†	4.6 ± 0.51	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00	4.0 ± 0.00

Table 9.1. Average accuracy and number of selected variables of *synthetic-1* artificial databases.

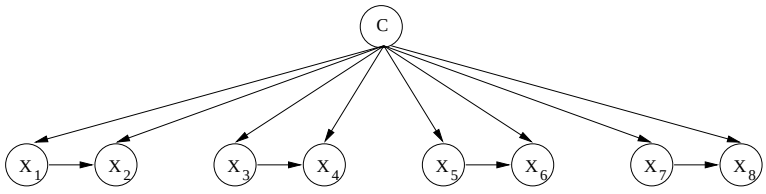
	synthetic-2											
	1000				5000				10000			
	acc.	±	*dev.*	*feat.* ± *dev.*	*acc.*	±	*dev.*	*feat.* ± *dev.*	*acc.*	±	*dev.*	*feat.* ± *dev.*
Naive Bayes	68.90	±	2.70	70.0 ± 0.00	72.39	±	1.01	70.0 ± 0.00	72.01	±	2.12	70.0 ± 0.00
TAN	88.20	±	2.52†	70.0 ± 0.00	97.56	±	1.00†	70.0 ± 0.00	98.73	±	0.73†	70.0 ± 0.00
kDB	81.90	±	3.61†	70.0 ± 0.00	91.34	±	1.11†	70.0 ± 0.00	98.52	±	0.76†	70.0 ± 0.00
Selective NB$_{fs}$	68.90	±	3.38	24.0 ± 0.00	72.40	±	2.42	24.00 ± 0.00	72.01	±	1.13	24.00 ± 0.00
Semi NB$_{fs}$	69.50	±	2.74	24.0 ± 0.00	83.39	±	3.13†	24.0 ± 0.00	81.40	±	1.25†	24.0 ± 0.00
TAN$_{fs}$	97.80	±	1.83†	24.0 ± 0.00	100.0	±	0.00†	24.0 ± 0.00	100.0	±	0.00†	24.0 ± 0.00
kDB$_{fs}$	99.50	±	0.92†	24.0 ± 0.00	100.0	±	0.00†	24.0 ± 0.00	100.0	±	0.00†	24.0 ± 0.00
Selective NB$_{ws}$	100.0	±	0.00†	4.1 ± 0.32	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00†	4.0 ± 0.00
Semi NB$_{ws}$	100.0	±	0.00†	4.4 ± 0.51	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00†	4.0 ± 0.00
TAN$_{ws}$	100.0	±	0.00†	4.1 ± 0.32	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00†	4.0 ± 0.00
kDB$_{ws}$	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00†	4.0 ± 0.00	100.0	±	0.00†	4.0 ± 0.00

Table 9.2. Average accuracy and number of selected variables of *synthetic-2* artificial databases.

9.4.1 Experimental results with synthetic databases

In order to test the quality of the the Bayesian classification models proposed in Sections 9.2 and 9.3, six artificial databases have been designed. The influence of the database size is proved, and each artificial domain is sampled to obtain 1000, 5000 and 10000 instances.

The redundant and irrelevant variables degrade the performance of naive Bayes, TAN and kDB in their original proposals. Thus, two databases with redundant and irrelevant variables are built to check the ability of the Bayesian classification models proposed to detect these kind of variables. In order to reject the irrelevant domain variables, *synthetic-1* is designed. It contains fifty discrete variables, among which only four are relevant to the class. All domain variables are randomly sampled in the $(3, 4, 5, 6)$ set. The class value is determined if (x_1, x_2, x_3, x_4) are closer by means of the Euclidean distance to $(9, 9, 9, 9)$ or to $(0, 0, 0, 0)$ points, that is, if $\sum_{i=1}^{4}(9 - x_i)$ is smaller than $\sum_{i=1}^{4} x_i$. Thus, only the first 4 variables take part in the setting of the class value and 46 are irrelevant with respect to the class. Following this method, 1000, 5000 and 10000 cases are simulated. *Synthetic-2* is generated in a similar way. However, to reject the irrelevant and redundant variables, each instance includes the 4 relevant variables, the 46 irrelevant variables and 20 clones of



C	
c_0	0.55
c_1	0.45

X_1, X_3, X_5, X_7		
C	c_0	c_1
s_0	0.6	0.4
s_1	0.4	0.6

X_2, X_4, X_6, X_8				
C, X_{i-1}	c_0s_0	c_0s_1	c_1s_0	c_1s_1
s_0	0.6	0.4	1.0	0.0
s_1	0.4	0.6	0.0	1.0

Fig. 9.16. Bayesian classifier simulated to obtain the *synthetic-3* datasets.

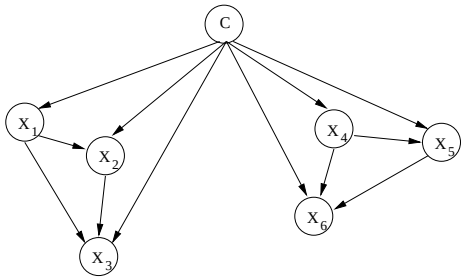
	synthetic-3 1000		5000		10000	
	acc. ± *dev.*	*feat.* ± *dev.*	*acc.* ± *dev.*	*feat.* ± *dev.*	*acc.* ± *dev.*	*feat.* ± *dev.*
Naive Bayes	60.80 ± 4.37	18.0 ± 0.00	60.56 ± 2.00	18.0 ± 0.00	60.97 ± 1.51	18.0 ± 0.00
TAN	91.70 ± 1.73†	18.0 ± 0.00	92.62 ± 1.64†	18.0 ± 0.00	92.92 ± 0.98†	18.0 ± 0.00
kDB	91.50 ± 3.00†	18.0 ± 0.00	92.62 ± 0.93†	18.0 ± 0.00	92.92 ± 0.45†	18.0 ± 0.00
Selective NB$_{fs}$	60.80 ± 3.89	15.3 ± 0.67	60.56 ± 2.64	18.0 ± 0.00	60.97 ± 1.26	18.0 ± 0.00
Semi NB$_{fs}$	70.40 ± 6.43†	15.4 ± 0.69	78.06 ± 5.63†	18.0 ± 0.00	76.84 ± 5.55†	18.0 ± 0.00
TAN$_{fs}$	78.60 ± 3.80†	15.5 ± 0.70	92.62 ± 1.27†	18.0 ± 0.00	92.92 ± 0.81†	18.0 ± 0.00
kDB$_{fs}$	64.60 ± 4.24	15.3 ± 0.67	60.92 ± 1.94	18.0 ± 0.00	61.41 ± 1.57	18.0 ± 0.00
Selective NB$_{ws}$	65.20 ± 4.23	3.3 ± 1.34	65.70 ± 2.55†	3.5 ± 1.43	69.40 ± 0.87†	2.0 ± 0.00
Semi NB$_{ws}$	87.00 ± 3.89†	7.3 ± 1.34	92.58 ± 0.90†	8.8 ± 0.79	85.88 ± 9.41†	6.4 ± 2.83
TAN$_{ws}$	64.50 ± 5.33	3.5 ± 1.18	66.68 ± 1.82†	4.1 ± 1.66	67.60 ± 3.40†	2.0 ± 0.00
kDB$_{ws}$	64.10 ± 7.35	2.5 ± 0.71	65.18 ± 3.87†	3.2 ± 1.03	67.78 ± 3.37†	2.0 ± 0.00

Table 9.3. Average accuracy and number of selected variables of the artificial databases simulated from *synthetic-3*.

X_1. This means that *synthetic-2* has 70 variables, 20 of which are redundant. The expectation is that the novel approaches to Bayesian classifiers reject the irrelevant and redundant variables and only take into account the four relevant variables. In this way, the predictive variables of *synthetic-1* and *synthetic-2* respect the naive Bayes classification scheme besides the inclusion of irrelevant variables in *synthetic-1* and the inclusion of irrelevant and redundant variables in *synthetic-2*. Tables 9.1 and 9.2 show the average accuracy attained (with its corresponding standard deviation) and the number of selected variables for each Bayesian classification model proposed.

A study of the statistically significant differences is carried out by means of the Mann-Whitney test Mann and Whitney, 1947. The symbol † denotes statistically significant improvements at the 0.01 confidence level with respect to naive Bayes.

In the *synthetic-1* database –Table 9.1–, the novel approaches outperform the results of naive Bayes, TAN and kDB in their original proposals with statistically significant differences when the database contains 1000 and 5000 cases. The filter and wrapper approaches to naive Bayes, TAN and kDB are able to detect the irrelevant variables and reject them as predictive variables



C	
c_0	0.55
c_1	0.45

	X_1, X_4		
C	C	c_0	c_1
	s_0	0.6	0.4
	s_1	0.4	0.6

	X_2, X_5			
C, X_{i-1}	c_0s_0	c_0s_1	c_1s_0	c_1s_1
s_0	0.6	0.4	0.8	0.2
s_1	0.4	0.6	0.2	0.8

	X_3, X_6							
C, X_{i-1}, X_{i-2}	$c_0s_0s_0$	$c_0s_0s_1$	$c_0s_1s_0$	$c_0s_1s_1$	$c_1s_0s_0$	$c_1s_0s_1$	$c_1s_1s_0$	$c_1s_1s_1$
s_0	0.6	0.4	0.6	0.4	0.8	0.2	0.8	0.2
s_1	0.4	0.6	0.4	0.6	0.2	0.8	0.2	0.8

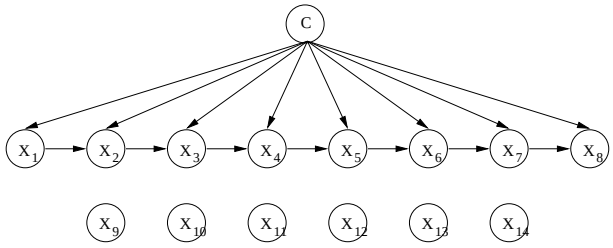
Fig. 9.17. Bayesian classifier simulated to obtain the *synthetic-4* datasets.

	synthetic-4											
	1000				5000				10000			
	acc.	±	*dev.*	*feat.* ± *dev.*	*acc.*	±	*dev.*	*feat.* ± *dev.*	*acc.*	±	*dev.*	*feat.* ± *dev.*
Naive Bayes	57.20	±	3.68	6.0 ± 0.00	60.56	±	1.96	6.0 ± 0.00	60.77	±	1.61	6.0 ± 0.00
TAN	72.40	±	2.87†	6.0 ± 0.00	71.12	±	2.02†	6.0 ± 0.00	70.65	±	1.61†	6.0 ± 0.00
kDB	70.20	±	5.41†	6.0 ± 0.00	70.90	±	2.51†	6.0 ± 0.00	70.39	±	1.08†	6.0 ± 0.00
Selective NB_{fs}	58.90	±	5.69	1.9 ± 0.32	61.29	±	2.38	3.3 ± 0.48	61.08	±	2.23	5.7 ± 0.48
Semi NB_{fs}	59.80	±	3.45	2.0 ± 0.00	62.16	±	2.07	3.3 ± 0.48	65.43	±	2.93†	5.7 ± 0.67
TAN_{fs}	59.80	±	4.28	2.0 ± 0.00	62.56	±	1.67	3.3 ± 0.48	70.18	±	0.98†	5.6 ± 0.51
$k\text{DB}_{fs}$	59.80	±	3.59	2.0 ± 0.00	62.16	±	2.07	3.3 ± 0.48	70.74	±	1.05†	6.0 ± 0.00
Selective NB_{ws}	59.40	±	2.65	2.9 ± 0.85	62.44	±	2.49	2.5 ± 0.71	63.21	±	1.79†	2.8 ± 0.92
Semi NB_{ws}	69.10	±	2.46†	4.2 ± 1.55	70.60	±	2.59†	5.6 ± 1.26	70.66	±	1.40†	6.0 ± 0.00
TAN_{ws}	60.00	±	4.71	2.7 ± 0.82	62.30	±	2.30	2.0 ± 0.00	62.95	±	2.35	2.7 ± 0.95
$k\text{DB}_{ws}$	58.70	±	5.56	2.4 ± 0.7	62.30	±	2.34	2.5 ± 0.53	62.93	±	1.07†	2.2 ± 0.42

Table 9.4. Average accuracy and number of selected variables of the artificial databases simulated from *synthetic-4*.

of the classification model. However, with 10000 cases, it seems that the irrelevant variables do not effect the naive Bayes performance and its accuracy is as good as in the case of the filter and wrapper approaches.

The results of the *synthetic-2* database –Table 9.2– are drastically influenced by the redundant variable and it produces a decrease in accuracy. In this database, the average results of 1000, 5000 and 10000 cases show similar behaviour. Although the filter approaches show statistically significant differences (in the seminaive Bayes with 5000 and 10000 cases and in the TAN and kDB) with respect to naive Bayes, they are not able to reject the redundant variables. However, the original proposals and the filter approaches to TAN and the kDB classification models detect the redundant variables. These



C , X_9, …, X_{14}	
c_0	0.55
c_1	0.45

	X_1	
C	c_0	c_1
s_0	0.6	0.4
s_1	0.4	0.6

	X_2, …, X_8			
C, X_{i-1}	c_0s_0	c_0s_1	c_1s_0	c_1s_1
s_0	0.6	0.4	1.0	0.0
s_1	0.4	0.6	0.0	1.0

Fig. 9.18. Bayesian classifier simulated to obtain the *synthetic-5* datasets.

	synthetic-5											
	1000				5000				10000			
	acc.	±	*dev.*	*feat.* ± *dev.*	*acc.*	±	*dev.*	*feat.* ± *dev.*	*acc.*	±	*dev.*	*feat.* ± *dev.*
Naive Bayes	63.80	±	2.96	14.0 ± 0.00	65.56	±	2.35	14.0 ± 0.00	67.14	±	0.95	14.0 ± 0.00
TAN	98.40	±	0.80	14.0 ± 0.00	98.44	±	0.35	14.0 ± 0.00	98.47	±	0.35	14.0 ± 0.00
kDB	98.40	±	1.11	14.0 ± 0.00	98.44	±	0.39	14.0 ± 0.00	98.47	±	0.26	14.0 ± 0.00
Selective NB$_{fs}$	57.80	±	4.77	1.0 ± 0.00	66.10	±	1.68	8.0 ± 0.00	67.35	±	1.35	8.0 ± 0.00
Semi NB$_{fs}$	57.80	±	4.11	1.0 ± 0.00	95.48	±	2.47	8.0 ± 0.00	96.05	±	2.15	8.0 ± 0.00
TAN$_{fs}$	57.80	±	2.78	1.0 ± 0.00	98.44	±	0.42	8.0 ± 0.00	98.47	±	0.40	8.0 ± 0.00
kDB$_{fs}$	57.80	±	4.06	1.0 ± 0.00	91.24	±	2.43	8.0 ± 0.00	91.86	±	2.92	8.0 ± 0.00
Selective NB$_{ws}$	68.90	±	4.71	2.3 ± 0.48	67.64	±	2.40	3.0 ± 0.94	70.49	±	1.55	2.3 ± 0.48
Semi NB$_{ws}$	98.40	±	1.01	8.5 ± 0.53	98.44	±	0.69	9.2 ± 0.79	98.47	±	0.40	8.8 ± 0.79
TAN$_{ws}$	70.00	±	4.47	2.1 ± 0.32	67.88	±	2.07	3.6 ± 0.96	70.92	±	1.58	2.3 ± 0.48
kDB$_{ws}$	69.10	±	2.66	2.3 ± 0.48	65.46	±	4.27	2.1 ± 0.32	70.96	±	1.31	2.0 ± 0.00

Table 9.5. Average accuracy and number of selected variables of the artificial databases simulated from *synthetic-5*.

classification models include arcs between the redundant variables forming a group. Hence, the deterioration of the average accuracy is only noticeable with 1000 cases. On the other hand, the wrapper approaches detect and reject redundant and irrelevant variables. Thus, the accuracy results reflect this fact with statistically significant differences regarding the naive Bayes model.

Once the ability to detect redundant and irrelevant variables is tested, the retrieval of probabilistic relationships between the predictive variables is studied. For this purpose, four artificial domains are built. These databases are built by means of Bayesian classifier construction and simulation. Figures 9.16, 9.17, 9.18 and 9.19 show the results of the Bayesian classification models studied. Nevertheless, it must be remarked that only *synthetic-3*, besides the predictive variables presented in Figure 9.16, contains ten clones of variable X_1.

Synthetic-3 and *synthetic-4* networks could be regarded as seminaive Bayes models where the supernodes are consist of two or three predictive variables, respectively. *Synthetic-5* follows a TAN structure with irrelevant variables and, finally, *synthetic-6* represents a kDB paradigm.

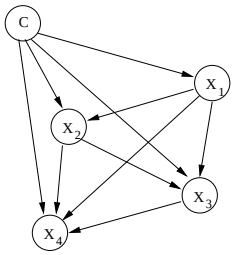
Tables 9.3, 9.4, 9.5 and 9.6 summarise the average accuracy results over the sampled databases by means of the *synthetic-3*, *synthetic-4*, *synthetic-5* and *synthetic-6* networks.

In the results of the *synthetic-3* and *synthetic-4* domains, an analysis of the differences between the average accuracy is run by the Mann-Whitney test Mann and Whitney, 1947. The symbol † in Tables 9.3 and 9.4 denotes statistically significant improvements at the 0.01 confidence level with respect to naive Bayes. As expected, the Bayesian classifiers which take into account the relationships between the variables outperform the results of naive Bayes.

Table 9.3 shows the power of the original TAN and kDB classifiers to properly classify data. In the filter approaches, the TAN model obtains the best results with statistically significant differences. Nevertheless, when the wrapper methods are considered, it is the seminaive Bayes classifier that attains the best accuracy results with statistically significant differences with respect to naive Bayes. The accuracy differences between the filter and wrapper approaches could be explained looking at the structures attained. Although the filter approach to TAN relates all the clones and, moreover, obtains the original structure as in Figure 9.16, the wrapper approaches to TAN and kDB select only two predictive variables joined by an arc. However, the wrapper approach to seminaive Bayes creates supernodes including all the predictive variables. Furthermore, it rejects the cloned variables of X_1. This fact explains the differences in accuracy of the wrapper approaches to Bayesian classifiers shown in Table 9.3.

The behaviour shown in Table 9.4 for the *synthetic-4* database is similar to the conduct appreciated in Table 9.3. In this database, the accuracy results are slightly lower but the statistically significant differences with respect to naive Bayes are maintained with 10000 cases. With 1000 and 5000 cases, only the original TAN and kDB and the wrapper seminaive Bayes are statistically better than naive Bayes. Moreover, the wrapper approach to seminaive Bayes attains better accuracy results than the other wrapper classifiers. Even though the results with the different sample size differ in the wrapper approaches, these differences become crucial in the filter approaches due to the number of variables selected. With 1000 and 5000 cases, only two and three predictive variables, respectively, are included in the classifier by filter methods. Thus, the average accuracy is slightly better than naive Bayes, without statistically significant differences. However, with 10000 cases, the filter approaches select almost all the predictive variables. Furthermore, the filter kDB is able to induce the structure of Figure 9.17 and the TAN classifier produces a sparse version of the structure.

Table 9.5 shows the accuracy results when working with the Bayesian classifiers proposed in Sections 9.2 and 9.3 over the databases simulated from the *synthetic-5* domain presented in Figure 9.5. As in the previous experiments, a deeper analysis of the results is performed using the Mann-Whitney test Mann and Whitney, 1947. In this case, as the structure of the simulated classifier follows a TAN model, the accuracy results of the filter and wrapper classifiers



C	
c_0	0.55
c_1	0.45

C	X_1	
	c_0	c_1
s_0	0.8	0.2
s_1	0.2	0.8

C, X_1	X_2			
	c_0s_0	c_0s_1	c_1s_0	c_1s_1
s_0	0.6	0.4	1.0	0.0
s_1	0.4	0.6	0.0	1.0

C, X_1, X_2	X_3							
	$c_0s_0s_0$	$c_0s_0s_1$	$c_0s_1s_0$	$c_0s_1s_1$	$c_1s_0s_0$	$c_1s_0s_1$	$c_1s_1s_0$	$c_1s_1s_1$
s_0	0.6	0.4	0.8	0.2	0.6	0.4	0.8	0.2
s_1	0.4	0.6	0.2	0.8	0.4	0.6	0.2	0.8

C, X_1, X_2, X_3	X_4							
	$c_0s_0s_0s_0$	$c_0s_0s_0s_1$	$c_0s_0s_1s_0$	$c_0s_0s_1s_1$	$c_0s_1s_0s_0$	$c_0s_1s_0s_1$	$c_0s_1s_1s_0$	$c_0s_1s_1s_1$
s_0	0.6	0.4	0.8	0.2	0.6	0.4	0.8	0.2
s_1	0.4	0.6	0.2	0.8	0.4	0.6	0.2	0.8
	$c_1s_0s_0s_0$	$c_1s_0s_0s_1$	$c_1s_0s_1s_0$	$c_1s_0s_1s_1$	$c_1s_1s_0s_0$	$c_1s_1s_0s_1$	$c_1s_1s_1s_0$	$c_1s_1s_1s_1$
2-9 s_0	0.6	0.4	0.8	0.2	0.6	0.4	0.8	0.2
s_1	0.4	0.6	0.2	0.8	0.4	0.6	0.2	0.8

Fig. 9.19. Bayesian classifier simulated to obtain the *synthetic-6* datasets.

are tested with respect to the accuracy of TAN as proposed in Friedman et al. (1997). No statistically significant improvements are noticed. Looking at the accuracy of the filter approaches, only the seminaive Bayes and the TAN with 5000 and 10000 instances are competitive with respect to the original TAN and kDB methods. When 1000 cases are considered, the filter approaches only add the X_1 variable, whereas when working with 5000 and 10000, cases all the relevant variables are included and the relationships between them taken into account. In the wrapper approaches, seminaive Bayes is competitive with respect to TAN and kDB due to the fact that all relevant variables are joined in a supernode. Naive Bayes in its original proposal and wrapper and filter approaches to selective naive Bayes attain an average accuracy lower than the one of the corresponding TAN and kDB models. This fact supports the assumption that, in this domain, models without dependencies between variables are not competitive enough. It must be noted that the irrelevant variables are rejected by both filter and wrapper methods.

In Table 9.6, the average accuracy and the standard deviation of the Bayesian classifier over the databases simulated from the *synthetic-6* domain are presented. As the network follows a kDB model as proposed in Sahami (1996), the Mann-Whitney test Mann and Whitney, 1947 is performed to look

	synthetic-6											
	1000				5000				10000			
	acc.	± *dev.*	*feat.*	± *dev.*	*acc.*	± *dev.*	*feat.*	± *dev.*	*acc.*	± *dev.*	*feat.*	± *dev.*
Naive Bayes	84.40	± 3.44	4.0	± 0.00	83.48	± 1.89	4.0	± 0.00	83.61	± 1.18	4.0	± 0.00
TAN	84.80	± 2.78	4.0	± 0.00	84.48	± 2.33	4.0	± 0.00	84.42	± 1.12	4.0	± 0.00
kDB	84.80	± 3.18	4.0	± 0.00	84.48	± 0.80	4.0	± 0.00	84.42	± 1.30	4.0	± 0.00
Selective NB$_{fs}$	84.40	± 4.10	3.8	± 0.42	83.48	± 1.93	4.0	± 0.00	83.61	± 1.32	4.0	± 0.00
Semi NB$_{fs}$	84.50	± 4.27	3.8	± 0.42	83.46	± 1.67	4.0	± 0.00	83.39	± 1.48	4.0	± 0.00
TAN$_{fs}$	84.80	± 3.37	3.8	± 0.42	84.48	± 1.53	4.0	± 0.00	84.42	± 1.19	4.0	± 0.00
kDB$_{fs}$	84.80	± 3.73	3.8	± 0.42	84.48	± 1.43	4.0	± 0.00	84.42	± 1.29	4.0	± 0.00
Selective NB$_{ws}$	83.70	± 4.77	3.1	± 0.87	81.16	± 3.10	1.9	± 1.20	81.59	± 2.08	2.4	± 1.50
Semi NB$_{ws}$	84.80	± 3.21	2.3	± 0.48	84.48	± 1.72	2.3	± 0.48	84.42	± 0.83	2.4	± 0.52
TAN$_{ws}$	83.80	± 2.52	3.1	± 0.87	81.68	± 1.84	2.2	± 1.31	81.68	± 1.48	2.3	± 1.42
kDB$_{ws}$	81.60	± 5.12	2.0	± 0.00	80.72	± 1.95	1.6	± 0.84	81.01	± 2.00	1.6	± 0.97

Table 9.6. Average accuracy and number of selected variables of the artificial databases simulated from *synthetic-6*.

for statistically significant improvements with respect to the kDB. Although there are no statistically significant improvements, the results of the filter approaches are competitive and slightly better than the results of the wrapper approaches.

The following set of conclusions briefly summarises the experimentation displayed in Tables 9.1, 9.2, 9.3, 9.4, 9.5 and 9.6:

- The wrapper approaches are able to detect and remove the irrelevant and redundant domain variables. However, the novel filter approaches proposed only reject the irrelevant domain variables and include the redundant variables in the classification model.
- When the problem domain contains probabilistic relationships between the variables, the accuracy results of the classifiers which take into account these relationships (seminaive Bayes, TAN and kDB) are similar in almost all the experiments. In fact, although each artificial dataset is designed to outperform a specific classifier, the results of seminaive Bayes, TAN and kDB are comparable and any classifier significantly obtains better results than the other models. Moreover, the accuracy results of naive Bayes with all the variables decrease. The filter approaches tend to obtain better accuracy results than the corresponding wrapper approaches. This fact seems to be produced by the short-sighted behaviour of the wrapper approaches, which only consider improving the accuracy at each step. As a result of this behaviour not all the relevant variables are selected to be part of the classifier and the accuracy results decrease.
- The number of cases influences the accuracy results and the number of selected variables. As the number of cases increases, the average accuracy increases slightly. With a small number of cases, the accuracy results of the Bayesian classifiers proposed are not reliable and surprising results are obtained. Furthermore, when there are probabilistic relationships among the domain variables, the filter approaches require a higher number of cases in the database in order to include enough variables and to take into account the relationships between pairs of variables. On the other hand,

the wrapper approaches require a high number of cases to reliably estimate the scoring measure to guide the search process.

9.4.2 Experimental results with UCI databases

In order to test the Bayesian classifiers presented in Sections 9.2 and 9.3 in real domains, a set of real-world databases from the UCI Repository Blake and Merz, 1998 are selected. These databases come from several fields and have different characteristics, numbers of instances and numbers of variables. Nevertheless, all the domain variables are discrete. The databases are as follows:

- The *breast* dataset was obtained from the University of Wisconsin Hospital, United States. The information collected is related to breast cytology for distinction between benign and malignant cancer.
- The *chess* database, specifically king+rook versus king+pawn, contains a set of chess board positions. The white side moves. The task is to predict whether white can win.
- The *lymphography* dataset was obtained from the University Medical Center, Institute of Oncology, Ljubljana, Slovenia. The aim is to distinguish between healthy patients and those with metastases or malignant lymphomata.
- The *mushroom* domain was drawn from the Audubon Society Field Guide to North American Mushrooms. The database includes instances of hypothetical samples corresponding to 23 species of gilled mushrooms of the Agaricus and Lepiota families. The mushroom must be labelled as edible or poisonous.
- The *splice* domain's real title is "primate splice-junction gene sequences with associated imperfect domain theory". The examples come from the Genbank. The problem consists on recognising the intron/exon boundaries and the exon/intron boundaries.
- the *vote* dataset was collected from the United States Congress in 1984. Given the voting records of the Congress members, each instance has to be classified as Democrat or Republican.

Table 9.7 sums up the characteristics of the databases, the number of variables, the number of instances, the states of the class and the number of missing values. As the Bayesian classifiers proposed require complete data, the missing values are handled replacing the unknown value with the mode given the class.

The average accuracy (and the corresponding standard deviation) of the 10-fold cross-validation process of the *breast*, *chess*, *lymphography*, *mushroom*, *splice* and *vote* databases is displayed in Tables 9.8, 9.9, and 9.10. The average number of selected variables (and the corresponding standard deviation) at each fold of the cross-validation is also presented.

	n. variables	*n. examples*	*n. classes*	*n. missing*
breast	9	699	2	16
chess	36	3196	2	0
lymphography	18	148	4	0
mushroom	22	8124	2	2480
splice	60	3190	3	0
vote	16	435	2	0

Table 9.7. Characteristics of the UCI Repository databases.

	breast		*chess*	
	acc. ± dev	feat. ± dev.	acc. ± dev	feat. ± dev.
Naive Bayes	97.27 ± 1.64	9.00 ± 0.00	87.52 ± 1.56	36.00 ± 0.00
TAN	95.41 ± 2.22	9.00 ± 0.00	92.18 ± 1.42†	36.00 ± 0.00
kDB	95.00 ± 1.68	9.00 ± 0.00	95.46 ± 0.88†	36.00 ± 0.00
Selective NB$_{fs}$	97.28 ± 0.99	9.00 ± 0.00	87.83 ± 0.63†	20.00 ± 0.50
Semi NB$_{fs}$	97.25 ± 1.11	9.00 ± 0.00	92.02 ± 2.22†	20.22 ± 0.44
TAN$_{fs}$	96.99 ± 1.87	9.00 ± 0.00	92.99 ± 1.08†	20.00 ± 0.00
kDB$_{fs}$	97.14 ± 2.03	9.00 ± 0.00	94.65 ± 1.57†	20.22 ± 0.44
Selective NB$_{ws}$	96.71 ± 1.69	5.11 ± 0.78	94.59 ± 1.50†	7.33 ± 2.12
Semi NB$_{ws}$	96.42 ± 1.85	5.67 ± 0.87	94.65 ± 0.99†	8.00 ± 2.65
TAN$_{ws}$	96.71 ± 1.56	4.89 ± 1.05	94.31 ± 1.01†	6.11 ± 0.93
kDB$_{ws}$	96.44 ± 2.40	4.89 ± 0.78	94.49 ± 0.83†	6.44 ± 2.19

Table 9.8. Average accuracy and average number of selected variables of the *breast* and *chess* databases.

A Mann-Whitney test Mann and Whitney, 1947 is carried out to analyse the differences of the Bayesian classifiers regarding the naive Bayes. The symbol † in Tables 9.8, 9.9 and 9.10 denotes a statistically significant improvement at 0.01 the confidence level.

In the databases with a small number of examples (*breast*, *lymphography* and *vote*), statistically significant improvements are not noticed (except in the original proposal and the wrapper version of the kDB model in the *vote* domain). In fact, in the *breast* and *lymphography* databases, naive Bayes attains the highest accuracy results. Moreover, in the *breast* domain, given the results of the filter approaches, all the predictive variables are relevant for classification purposes. Hence, despite the feature reduction of the wrapper methods, the average accuracy is slightly lower than the naive Bayes accuracy but without statistically significant differences.

In the *lymphography* domain, a significant feature reduction is performed by the filter and the wrapper methods, but this reduction does not attain a statistically significant improvement in accuracy. In fact, this feature reduction implies a decrease in accuracy. In the *vote* domain, although the wrapper approaches select a small number of variables and accuracy increases, only

	lymphography		*mushroom*	
	acc. ± dev	feat. ± dev.	acc. ± dev	feat. ± dev.
Naive Bayes	85.33 ± 9.93	18.00 ± 0.00	95.38 ± 0.52	22.00 ± 0.00
TAN	83.55 ± 6.82	18.00 ± 0.00	99.98 ± 0.04†	22.00 ± 0.00
kDB	79.31 ± 11.15	18.00 ± 0.00	100.0 ± 0.00†	22.00 ± 0.00
Selective NB$_{fs}$	75.75 ± 10.93	3.22 ± 0.83	95.32 ± 0.66	21.00 ± 0.00
Semi NB$_{fs}$	73.60 ± 8.70	3.44 ± 1.01	99.67 ± 0.53†	21.00 ± 0.00
TAN$_{fs}$	72.21 ± 9.22	3.00 ± 0.71	99.97 ± 0.05†	21.22 ± 0.44
kDB$_{fs}$	74.60 ± 14.61	3.33 ± 0.71	99.84 ± 0.27†	21.22 ± 0.44
Selective NB$_{ws}$	74.13 ± 10.72	4.67 ± 2.00	99.69 ± 0.17†	3.78 ± 0.67
Semi NB$_{ws}$	80.79 ± 8.76	4.33 ± 2.00	99.85 ± 0.11†	5.56 ± 1.01
TAN$_{ws}$	80.88 ± 14.44	4.56 ± 2.13	99.85 ± 0.15†	5.33 ± 0.87
kDB$_{ws}$	75.84 ± 12.27	4.89 ± 2.15	99.79 ± 0.16†	4.33 ± 1.32

Table 9.9. Average accuracy and average number of selected variables with of *lymphography* and *mushroom* databases.

	splice		*vote*	
	acc. ± dev	feat. ± dev.	acc. ± dev	feat. ± dev.
Naive Bayes	95.29 ± 0.56	60.00 ± 0.00	90.16 ± 4.42	16.00 ± 0.00
TAN	94.95 ± 1.56	60.00 ± 0.00	94.02 ± 3.11	16.00 ± 0.00
kDB	87.74 ± 1.28	60.00 ± 0.00	95.62 ± 2.63†	16.00 ± 0.00
Selective NB$_{fs}$	96.11 ± 1.27	35.00 ± 1.94	89.89 ± 2.56	14.00 ± 0.00
Semi NB$_{fs}$	96.07 ± 1.43	34.75 ± 1.23	92.65 ± 2.86	14.00 ± 0.00
TAN$_{fs}$	96.05 ± 1.61	34.67 ± 1.00	94.26 ± 3.42	14.00 ± 0.00
kDB$_{fs}$	95.77 ± 1.08	34.56 ± 1.13	93.12 ± 2.25	14.00 ± 0.00
Selective NB$_{ws}$	94.86 ± 1.17	13.00 ± 2.00	94.94 ± 2.90	2.89 ± 0.93
Semi NB$_{ws}$	92.76 ± 2.11	10.22 ± 2.82	94.95 ± 2.23	3.89 ± 0.78
TAN$_{ws}$	94.92 ± 1.78	10.56 ± 2.35	94.71 ± 2.52	2.89 ± 1.05
kDB$_{ws}$	95.14 ± 1.14	11.67 ± 3.08	95.64 ± 3.31†	2.78 ± 0.67

Table 9.10. Average accuracy and average number of selected variables with of *splice* and *vote* databases.

the kDB model shows statistically significant differences with respect to naive Bayes. However, the average accuracy of the wrapper methods is higher than the average accuracy of the corresponding filter models.

The databases with a high number of variables (*chess*, *mushroom* and *splice*) behave differently. In the *chess* domain, besides feature reduction, an accuracy increase by the use of filter (except the selective naive Bayes) and wrapper methods is attained with statistically significant differences. Furthermore, the accuracy of the wrapper approaches is higher than the accuracy of the corresponding filter methods with a smaller number of selected variables.

The *mushroom* database results are similar to the *chess* results. Naive Bayes obtains the lowest accuracy results with statistically significant differ-

ences with respect to the other classification models except the filter approach to selective naive Bayes. Apart from the small number of variables selected by the wrapper approaches (about 5 of 22), the average accuracy of the wrapper methods is competitive with the average accuracy of the filter algorithms.

Finally, statistically significant differences are not noticed in the *splice* dataset. In spite of the feature reduction of the filter and wrapper approaches (from 60 to around 35 and 12 variables, respectively), the naive Bayes with all the domain variables attains a competitive accuracy regarding the wrapper methods.

Given the results of the real-world domains, the wrapper methods seem to be good inducers if the number of instances is large enough. In this way, a feature reduction coupled with an accuracy competitive at least with respect to the naive Bayes model is obtained. However, if the number of instances is small, the wrapper approaches tend to behave almost randomly, which seems to be related to the unreliability of the internal cross-validation method to guide the search process.

The naive Bayes with all the variables reaches competitive accuracy results in almost all real domains tested (except in the *chess* and *mushroom* databases). This fact could be related to the non-existence of dependencies between the domain variables. However, looking at the accuracy results, it could also be related to the number of cases, not large enough to detect those dependencies. In fact, naive Bayes is surpassed by the classification models proposed in the *chess* and *mushroom* databases, which have a considerable number of cases.

It must be remarked that the wrapper approaches require a high computational cost. Nevertheless, the extra cost does not produce a significant improvement of accuracy with respect to the filter methods.

10

Conclusions and future work

10.1 Conclusions

Novel filter and wrapper approaches to learning Bayesian classification models are proposed following the ideas of feature subset selection. The introduction of novel filter and wrapper approaches is suggested by the experimentation presented in Section 9.1.1. The assumption of a Bayesian classifier with a high likelihood performs better in terms of accuracy than another with small likelihood is tested with disappointing results. No relationship between the likelihood and the accuracy could be noticed. Nevertheless, a kind of relationship between the mutual information and the accuracy can be clearly appreciated.

The filter approaches to Bayesian classifier induction are based on the results of Pardo (1997). This result proves that $2NI(X_i, C)$ asymptotically follows a χ^2 probability distribution with $(r_i - 1)(r_0 - 1)$ degrees of freedom where N is the number of cases, X_i is a domain variable, C is the class, r_i is the states number of X_i and r_0 is the states number of the class. Thus, given the mutual information of a predictive variable and the class value, a $\chi^2_{(r_i-1)(r_0-1);1-\alpha}$ test could be performed to check the former property. This outcome is adapted to the characteristics of selective naive Bayes, seminaive Bayes, TAN and kDB Bayesian classifiers.

The wrapper approaches to Bayesian classifier induction are performed like a search process in which accuracy guides the search through the search space. The space of Bayesian classifier structures is greedily explored following the characteristics of each classification model.

The novel filter and wrapper approaches proposed are applied over a set of artificial and real datasets.

The aim of the design of the artificial domains is two-fold: to check the ability of the classifiers to detect irrelevant and redundant variables, and to explore the behaviour of the Bayesian classifiers which take into account the relationships between the domain variables. Moreover, the differences between the filter and wrapper approaches and the influence of the number of cases are

examined. On the other hand, the real datasets come from the UCI repository Blake and Merz, 1998.

According to the results of the artificial datasets, some interesting characteristics of the novel filter and wrapper approaches can be drawn. The requirement of a classifier more powerful than naive Bayes to reflect the relationships among the variables can be appreciated in the accuracy results. Whereas the filter methods presented are only able to detect the irrelevant domain variables, the wrapper methods presented notice the existence of irrelevant and redundant variables. Nevertheless, the wrapper classifiers do not include all the relevant variables. The seminaive Bayes, TAN and kDB models behave similarly and their accuracy results are comparable when the domain contains dependencies among the variables. The number of cases have a strong influence on both filter and wrapper classifiers. With a small number of cases, surprising results are obtained: filter approaches are not able to detect all the relevant variables and the score measure of the wrapper methods is unreliable.

The results of the UCI datasets supports the importance of the number of cases. In these real domains, the wrapper methods are good classifiers if the number of instances is big enough. This means that the election of accuracy as the search process guide is feasible.

The competitive accuracy results of naive Bayes in the real datasets is frustrating. However, the novel approaches obtain higher accuracy in the datasets with a large number of cases. Thus, the good performance of naive Bayes seems to be related to the small number of cases, not enough to detect dependencies between the variables (in the case of filter methods) and to make accuracy reliable as the search process guide (in the case of wrapper methods). Moreover, the belief that UCI datasets are pre-processed exits in the machine learning field Kohavi, 1995. This way, the UCI datasets would not have *irrelevant* variables and would only contain relevant variables for classification purposes. This assumption support the good performance of the naive Bayes classifier.

10.2 Future work

The Bayesian classification models are developing tools in the machine learning field. The novel approaches to supervised classification presented in this work have some limitations. Several interesting improvements to overcome the limitations could be initiated.

The empirical results to prove the relationship between accuracy and likelihood are not so good as expected. Nevertheless, this empirical experimentation to prove any relationship between accuracy and other scoring functions will be extended. Due to its use in the Bayesian networks field, the conditional log-likelihood is a good candidate. Furthermore, the relationship between accuracy, Brier score, ROC curves and calibration would be explored.

With respect to the filter approaches, the filter method proposed to intrinsically select the variables to build the classifier is not able to detect the redundant variables. The literature is full of filter measures. An intensive and exhaustive experimentation could reach a filter measure which clearly selects only the relevant domain variables rejecting the irrelevant and the redundant features.

The wrapper approaches to Bayesian classification models proposed are greedily guided by accuracy. The experimentation performed proves that this framework could be short-sighted due to the myopic point of view of the greedy search. Thus, in order to avoid the local optima, the adaptation of EDA algorithms to perform Bayesian classifier induction is being developed.

Moreover, the use of accuracy as a search guide could be short-sighted itself. This score measure only looks for the best accuracy without taking into account other interesting criteria like the Brier score or the likelihood of the classifier. Thus, a multi-criteria search is an interesting research work line.

Part IV

Applications in Biomedicine

11

Survival of cirrhotic patients treated with TIPS

The *transjugular intrahepatic portosystemic shunt* (TIPS) placement is a non-surgical technique to prolong the life expectancy and life quality of cirrhotic patients. Nevertheless, not all the patients survive a fixed time period. In this chapter, by means of a dataset collected by the Clínica Universitaria de Navarra, Spain, the identification of risk factors is analysed.

The chapter is organised as follows. In Section 11.1 the medical problem is introduced and the election of the time period is explained. Section 11.2 presents the dataset provided by the Clínica Universitaria de Navarra and the medical variables which compose it. Finally, Section 11.3 shows a thorough empirical experiment.

The work presented in this chapter is performed in collaboration with M.L. Merino, physician of the Servicio Vasco de Salud-Osakidetza, and it is an adaptation of Blanco et al. (2005).

11.1 Introduction

In the western world, 90% of the cases of portal hypertension are caused by liver cirrhosis. Portal hypertension is a pathological increase in portal venous pressure. This increase produces portosystemic collaterals that divert the portal blood flow from the liver to systemic circulation. As a result, metabolic and hemodynamic disorders responsible for most portal hypertension complications are originated.

Portal hypertension has serious consequences, i.e. gastro-oesophageal varices, hepatic encephalopathy, hypersplenism, and ascites. The bleeding originated by gastro-oesophageal varices is a significant cause of mortality (approximately 30%-50% at the first bleeding) Bornman et al., 1994; Saunders et al., 1981.

The *transjugular intrahepatic portosystemic shunt* (TIPS) is a non-surgical method resulting in decompression of the portal system. A prosthesis is placed between the portal and the suprahepatic veins by means of the angiographic

method. In spite of the number of studies carried out, the relationship between TIPS and the survival of treated patients is almost unknown.

Our medical staff identified a subgroup of patients which died within six months after TIPS placement whereas the rest of the patients survived for long periods. No risk factors have been determined to distinguish between both subgroups.

The choice of a six-month period is based on critical reasons. Medical factors like stenosis of the shunt and rebleeding would complicate the analysis. Moreover, the medical study does not show important variations in mortality after the first six months after TIPS placement. Another study Malinchoc et al., 2000, which also tries to identify the subgroup of patients who died within a period of time after TIPS placement, shortens this period to three months. However, a similar study Chalasani et al., 2000 extends the period of analysis to 36 months in order to reach a reliable clinical index to predict the consequences of TIPS placement.

Traditionally, Pugh's modification of the Child-Turcotte classification (referred to as the Child-Pugh classification) has been used to assess risk in patients undergoing portosystemic shunt surgery Pugh et al., 1973. Despite its traditional use to assess the seriousness of liver disease, it has inherent problems when applied to patients undergoing TIPS. It cannot be used to predict the survival of the patients within a certain period of time. The several difficulties and inaccuracies when using Child's classification have been detailed in Conn, 1981.

Since this work focusses on the prediction of survival within six months after TIPS placement coupled with reliability of the models and the satisfaction of the medical staff, Bayesian classification models are applied to the dataset of patients. Filter and wrapper approaches to selective naive Bayes, seminaive Bayes, TAN and k dependence Bayesian classifier presented in Chapter 9 are carried out to identify the risk factors after TIPS placement.

11.2 Patients: cases and variables

From May 1991 to September 1998, 134 patients suffering from liver cirrhosis underwent TIPS placement at the Clínica Universitaria de Navarra, Spain. In all cases, the diagnosis of cirrhosis was based on liver histology. However, only 127 patients were included in the study due to medical reasons.

The indications for TIPS placement were prophylaxis of rebleeding (68 patients), refractory ascites (28 patients), prophylaxis of bleeding (11 patients), acute bleeding refractory to endoscopic and medical therapy (10 patients), portal vein thrombosis (9 patients) and Budd-Chiari syndrome (1 patient).

The prospective study includes 107 patients because 20 patients underwent liver transplants within the first six months after TIPS placement. Bearing in mind that the aim of the study is to predict survival within the first six months after TIPS placement, the follow-up of these patients was rejected

History finding attributes:		
Age	Gender	Height
Weight	Etiology of cirrhosis	Indication of TIPS
Bleeding origin	Number of bleedings	Prophylactic therapy with propranolol
Previous sclerotherapy	Restriction of proteins	Number of hepatic encephalopathies
Type of hepatic encephalopathy	Ascites intensity	Number of paracenteses
Volume of paracenteses	Dose of furosemide	Dose of spironolactone
Spontaneous bacterial peritonitis	Kidney failure	Organic nephropathy
Diabetes mellitus		
Laboratory finding attributes:		
Hemoglobin	Hematocrit	White blood cell count
Serum sodium	Urine sodium	Serum potassium
Urine potassium	Plasma osmolarity	Urine osmolarity
Urea	Plasma creatinine	Urine creatinine
Creatinine clearance	Fractional sodium excretion	Diuresis
GOT	GPT	GGT
Alkaline phosphatase	Serum total bilirubin (mg/dl)	Serum conjugated bilirubin (mg/dl)
Serum albumin (g/dl)	Platelets	Prothrombin time (%)
Partial thrombin time	PRA	Proteins
FNG	Aldosterone	ADH
Dopamine	Norepinephrine	Epinephrine
Gammaglobulin		
CHILD score		
PUGH score		
Doppler sonography:		
Portal size	Portal flow velocity	Portal flow right
Portal flow left	Spleen length (cm)	
Endoscopy:		
Size of oesophageal varices	Gastric varices	Portal gastropathy
Acute hemorrhage		
Hemodynamic parameters:		
Arterial pressure (mm Hg)	Heart rate (beats/min)	Cardiac output (l/min)
Free hepatic venous pressure	Wedged hepatic venous pressure	Hepatic venous pressure gradient (HVPG)
Central venous pressure	Portal pressure	Portosystemic venous pressure gradient
Angiography:		
Portal thrombosis		

Table 11.1. Attributes of the study database for TIPS placement.

on the day of the transplant. The inclusion of patients who underwent liver transplants might have influenced the survival results. The survival prediction of the Bayesian classification models might have been modified by the surgical mortality related to transplantation. On the other hand, transplant patients may live longer than patients who do not undergo TIPS. Moreover, according to Malinchoc et al. (2000), survival in patients who undergo transplantation is significantly higher than those who do not undergo transplantation.

The database contains 77 clinical findings –see Table 11.1– for each patient. These 77 attributes were measured before TIPS placement. A new binary variable is created, called *vital-status*, which reflects whether or not the patients died within the first six months after the placement of TIPS. The values for this variable correspond to both classes of the problem. Within the first six months after TIPS placement, 33 patients died and 74 survived for a longer period, thus reflecting that the utility and consequences of TIPS were not the same for all the patients. Hence, our objective is to build Bayesian classification models that discriminate between these two subgroups of patients.

The study was approved by the local ethics committee, and informed oral consent was obtained from all patients.

11.3 Experimental results

The aim of this work is to reach the highest accuracy with feature reduction when identifying the subgroup of patients surviving six months after TIPS placement, coupled with the reliability of the classifiers and satisfaction of the medical staff with the Bayesian classification models. The empirical study is focussed on the accuracy of the wrapper and filter approaches to Bayesian classification models proposed in Chapter 9. However, the number of features selected and, in the case of the wrapper approaches, the number of evaluations required are also reported. In order to perform a deeper study, the ROC curves of the proposed classifiers are presented. In order to validate the Bayesian classification models, a *leave-one-out* cross-validation is performed. When a wrapper approach is used, a 5-fold cross-validation is performed as the internal accuracy estimation which guides the search process for the best model.

The parameters of all Bayesian classification models proposed are estimated applying the Laplace correction Laplace, 1814 to their maximum likelihood parameter estimations. The α parameter of the filter approaches proposed is fixed at $\alpha = 0.01$. The k of all the kDB classifiers is fixed at $k = 3$.

The study database contains missing data and continuous variables. The Bayesian classifiers presented in Chapter 9 are implemented to manage complete discrete databases. Therefore, the imputation of the missing values is carried out replacing the missing value by the mean (when the variable is continuous) or the mode of the variable (when the variable is discrete), conditioned to the value of the class. After imputation, continuous variables of the dataset are discretised by means of the *equal frequency* Cattlet, 1991 algorithm into two intervals.

Table 11.2 shows the estimated average accuracy (and its standard deviation) and the number of features of the final classifier induced for the Bayesian classifiers presented. The wrapper approaches have a related computational cost. Thus, the number of classification models evaluated is also displayed. The results support the fact that not all the features are needed in order to learn an accurate classification model, that is, a feature reduction process could increase the accuracy.

In order to compare statistical differences in the behaviour of the different Bayesian classification models, the Mann-Whitney test Mann and Whitney, 1947 is performed for all the Bayesian classifiers with respect to the naive Bayes. In spite of the non-existence of statistically significant differences, the filter approaches of all the Bayesian classification models improve the estimated accuracy in relation to the original methods and the wrapper approaches (except the semi naive Bayes classifier). The estimated accuracies achieved are competitive with previous studies Malinchoc et al., 2000; Chalasani et al., 2000; Inza et al., 2001b; Merino, 2004, and are considered 'acceptable' by physicians.

In order to select a classifier, researchers must bear in mind not only the estimation of accuracy, but also the time complexity or the number of

	accuracy	*n. features*	*n. evaluations*
Naive Bayes	88.78 ± 3.06	77	
TAN	88.78 ± 3.06	77	
kDB	88.78 ± 3.06	77	
Selective NB_{fs}	93.46 ± 2.40	11	
Semi NB_{fs}	93.46 ± 2.40	11	
TAN_{fs}	93.46 ± 2.40	11	
$k\text{DB}_{fs}$	93.46 ± 2.40	11	
Selective NB_{ws}	92.52 ± 2.55	3	302
Semi NB_{ws}	93.46 ± 2.40	5	1111
TAN_{ws}	90.65 ± 2.82	4	395
$k\text{DB}_{ws}$	92.52 ± 2.55	4	395

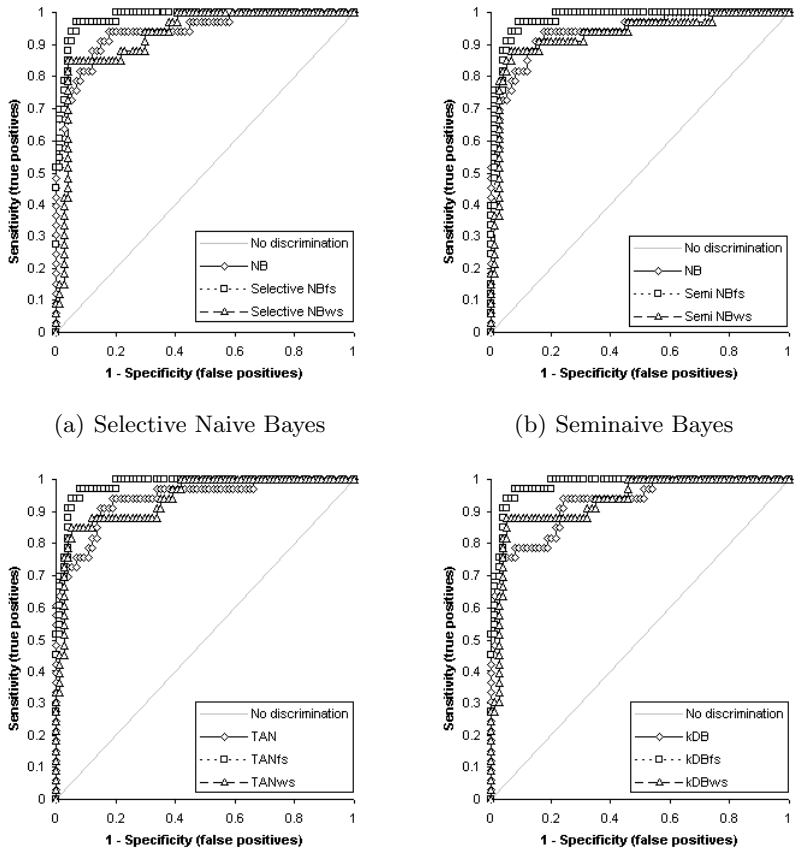
Table 11.2. Average results: estimated accuracy, number of features of the classifier induced and number of evaluations required.

features included in the models. To analyse the results more thoroughly, the ROC curves of each Bayesian classifier proposed are drawn. Widely used in biomedical applications, the ROC analysis provides tools to select classifiers independently of the cost context or the class distribution. A ROC curve supplies a concise graphic depiction of the overall performance of a classifier plotting the true positive rate against the false positive rate for the different possible cut-points.

Figure 11.1 shows the ROC curves when the probability threshold to assess the *vital-status* of a case as *alive* changes. Bayesian classification models with the same underlying structure are displayed together in such a way that the filter and wrapper approaches for each classification model are in the same figure. Looking at the ROC curves, it must be noted that the filter approaches of all the Bayesian classification models proposed reach a slightly higher sensibility and specificity than the wrapper approaches and the classifiers without feature selection.

Aside from the accuracy and the ROC curves, the calibration concept could be taken into account when selecting a classification model. The Brier score, in combination with other scoring function such as accuracy or ROC curves, could be a useful measure to select the 'best' classifier for a problem domain.

The Brier scoring measure of each final classification model is used to estimate the classifier calibration. It is also calculated by means of a *leave-one-out* cross-validation process. Table 11.3 shows the average Brier score (and its standard deviation) for the Bayesian classification models proposed. In order to compare statistical differences in the Brier score of the classifiers proposed, the Mann-Whitney test Mann and Whitney, 1947 is performed for all the Bayesian classifiers with respect to the naive Bayes. The symbol † denotes a statistically significant improvement at the 0.01 confidence level with respect to naive Bayes. The results of the newly proposed approaches are better than



(a) Selective Naive Bayes

(b) Seminaive Bayes

(c) TAN

(d) kDB

Fig. 11.1. ROC curves for the proposed Bayesian classification models when the probability threshold to asses the *vital-status* changes.

the naive Bayes outcome with statistical significant differences. Looking at the Brier score values, the filter approaches of the Bayesian classifiers could be considered more calibrated than the models obtained by wrapper methods.

Table 11.4 shows the variables included in the final Bayesian classification model built over the entire dataset. Note that the filter methods proposed reach the same eleven variables.

Although wrapper approaches are time-consuming algorithms, the dimension reduction achieved is significant in contrast to the original algorithms and the filter approaches. Physicians note that the dimension reduction affects data acquisition, reducing the extra costs of the medical tests (limiting the number of tests, the cost is automatically reduced) and the number of

	Brier score
Naive Bayes	0.1016 ± 0.281
TAN	0.1034 ± 0.288
kDB	0.0990 ± 0.280
Selective NB$_{fs}$	$0.0529 \pm 0.193^{\dagger}$
Semi NB$_{fs}$	$0.0553 \pm 0.189^{\dagger}$
TAN$_{fs}$	$0.0532 \pm 0.198^{\dagger}$
kDB$_{fs}$	$0.0536 \pm 0.196^{\dagger}$
Selective NB$_{ws}$	$0.0705 \pm 0.185^{\dagger}$
Semi NB$_{ws}$	$0.0597 \pm 0.149^{\dagger}$
TAN$_{ws}$	$0.0834 \pm 0.211^{\dagger}$
kDB$_{ws}$	$0.0780 \pm 0.206^{\dagger}$

Table 11.3. Average Brier score and its standard deviation for the proposed Bayesian classifiers proposed.

	Filter	Sel NB$_{ws}$	Semi NB$_{ws}$	TAN$_{ws}$	kDB$_{ws}$
History finding attributes:					
Etiology of cirrhosis			X		
Indication of TIPS		X	X	X	X
Spontaneous bacterial peritonitis	X				
Laboratory finding attributes:					
Plasma osmolarity	X	X	X	X	X
Creatinine clearance	X				
Serum albumin (g/dl)	X				
Aldosterone	X				
ADH	X				
Epinephrine				X	
PUGH score	X				
Doppler sonography:					
Portal flow left	X				
Portal flow velocity			X		
Spleen length (cm)	X				
Hemodynamic parameters:					
Arterial pressure (mm Hg)	X				
Heart rate	X	X	X	X	X
Hepatic venous pressure gradient (HVPG)					X

Table 11.4. List of variables included in the Bayesian classifiers.

invasive and painful medical techniques (endoscopy, angiography and hemodynamic tests in the majority of the Bayesian classifier models), which are not required. Most of the variables selected are related to the patients' medical histories and laboratory findings. While this subgroup of variables (histories and laboratory findings) is easily obtained without any significant inconvenience for the patients, the inconvenience for the patients increase with the rest of the medical tests, particularly with the hemodynamic test, where a catheter is introduced to examine the state of the vein. In spite of the requirement of a hemodynamic test to obtain the value of the HVPG variable, it is only introduced by the kDB$_{ws}$ classifier.

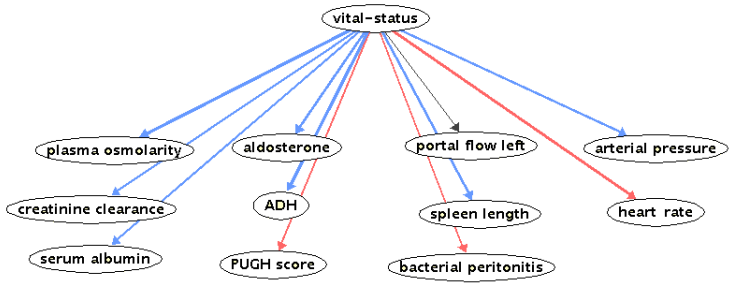


Fig. 11.2. Selective naive Bayes structure achieved by the filter approach.

Comparing this analysis with a previous study of the same database presented in Inza et al. (2001b), the variables selected by the Bayesian classifiers are not exactly the same. However, the medical variables selected in this study are interrelated with the selected variables in Inza et al. (2001b).

Figures 11.2, 11.3 and 11.4 show the structure of the Bayesian classification models achieved with some of the algorithms presented. The different types of arrows denote the kind of probabilistic relations between the *vital-status* class variable and the predictive variables. A red link represents positive influence between the parent and the son variable, that is, with a major value of the parents, the probability of higher values in the son variable increases. A blue arrow represents negative influence between the parents and the son, that is, with a major value of the parents, the probability of higher values in the son variable decreases. A purple link represents unknown probabilistic influence which cannot be formalised. The black arrows show irrelevant influence. The arrows' width describes the strength of the probabilistic influence. More information about the interpretation of the links could be read in Lacave, 2002.

Figure 11.2 shows the Bayesian classification structure obtained by a filter selective naive Bayes. Physicians note that the variables selected to induce filter-like classifiers are related to all the stages of a cirrhotic disease, from compensated cirrhosis to ascites. In spite of the ability of other filter approaches to include relationships between the problem variables, the attained model structures are similar to the classifier exposed in Figure 11.2. This simplification of the models takes place because of the weakness of the relationships between the predictive variables. This fact seems to be related to the dimension of the data, where few examples support the parameters involved in the relations.

Figure 11.3 presents the selective naive Bayes structure obtained by the wrapper approach. The *heart rate* feature is related to liver failure and the *plasma osmolarity* variable, to renal failure. The *indication of TIPS* variable states the reason for TIPS placement. Furthermore, in order to add the pre-

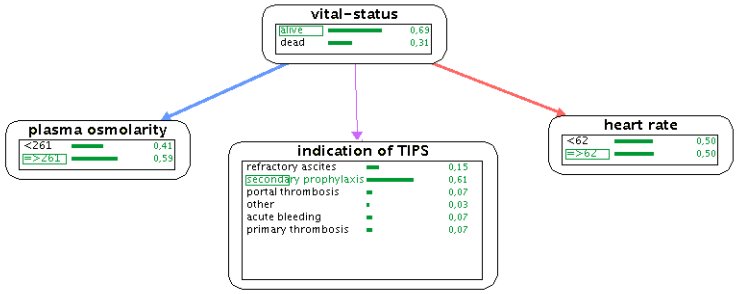


Fig. 11.3. Selective naive Bayes structure obtained by the wrapper approach.

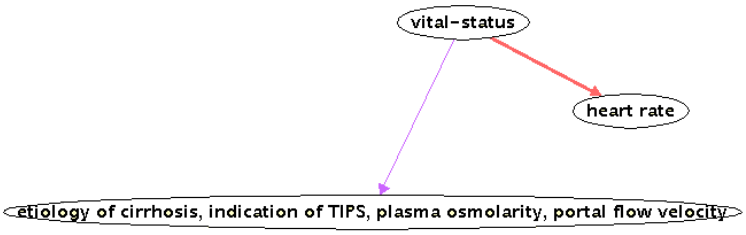


Fig. 11.4. Seminaive Bayes structure obtained by the wrapper approach.

vious knowledge about the underlying distribution of the dataset, the prior marginal probability distribution of each selected variable is displayed.

In Figure 11.4, the seminaive Bayes classifier obtained by the wrapper approach is presented. The features selected seem to be related to the initial stages of cirrhosis, where *heart rate* and *portal flow velocity* are connected with the vasoconstrictors secreted by the liver. Predictive variables that form the supernode are indirectly related: the join of these variables seems to be caused by the small number of cases in relation to the number of variables.

Physicians acknowledge the non-occurrence of 'subjective' variables (where the value of the variable is determined by the medical staff) in the wrapper Bayesian classification models. This means that the final results of the Bayesian classifiers are not influenced by the point of view of the physicians unlike the traditionally-used Child-Pugh score, which is a collection of five variables, two of which are based on medical opinion.

In relation to the set of *a posteriori* probability distributions, the conditional probabilities related to Bayesian classification models roughly assert the previous medical knowledge about cirrhosis. When the Bayesian classifiers are presented to the medical staff, physicians notice an improvement in comprehensibility and simplicity among the models induced by filter and wrapper approaches with respect to the original models induced by the whole set of variables. In other words, the dimension reduction, carried out in the filter

and wrapper approaches, reduces the complexity of the final classifiers. This fact provides a useful classification tool that could easily be used in everyday medical practice.

12

Identifying the oesophageal carcinoma type

The oesophageal carcinoma has different stages. To correctly identify these stages is a crucial task in order to give each patient the adequate therapy. By means of a collected dataset from the Netherlands Cancer Institute, Antoni van Leeuwenhoekhuis, an automatic way to discover the carcinoma stage is proposed.

The chapter is organised as follows. Section 12.1 briefly introduces the oesophageal carcinoma domain. In Section 12.2, the network which models the domain and the dataset provided by the Netherlands Cancer Institute are presented. The chapter concludes with the empirical results in Section 12.3.

The work is performed in collaboration with L.C. van der Gaag, leader of the Decision Support Systems group at Utrecht University and it is an extension of Blanco et al. (2004c).

12.1 Introduction

An *oesophageal carcinoma* is usually produced by a lesion of the oesophageal wall. Frequently, this lesion is a consequence of frequent reflux or is related to smoking and drinking habits. Due to an oesophageal carcinoma, a patient has difficulty swallowing food and could lose weight.

The difficulty to have a normal life depends on the extent of the carcinoma, which is conditioned by characteristics such as its location in the oesophagus, its histological type, length and macroscopic shape. Moreover, the carcinoma usually invades the oesophagus wall and it could invade, depending on the tumour location in the oesophagus, the adjacent organs such as the trachea and bronchi, the heart, the mediastinum, or the diaphragm.

A lymphatic metastasis in the lymph nodes and a haemotagenous metastasis in the lungs and the liver could appear as the carcinoma grows. The depth of invasion and extent of the metastasis are summarised in the stage of the carcinoma. The stage of the carcinoma and the physical condition of

the patient influence the life expectancy. At the same time, these characteristics are indicators of the effects and complications to be expected from the different possible therapies.

Medical diagnostic tests are performed to check the stage of the carcinoma. Usually, several biopsies of the primary tumour, gastroscopic and endosonographic examination of the oesophagus and the CT-scan of the chest and liver of the patient are carried out.

The Netherlands Cancer Institute, Antoni van Leeuwenhoekhuis, is a centre specialised in the treatment of cancer patients. Every year about eighty patients receive oesophageal carcinoma treatment at the centre. A standard protocol is performed in order to assign a therapy to these patients. Based on this protocol, 75% of these patients show a favourable response to their therapy. However, the remaining 25% develop serious complications as a result of the therapy.

Although detecting the oesophageal tumour in a patient is relatively straightforward, the detection of the stage of the carcinoma is a hard task and different therapies are related to each stage of the oesophagus carcinoma. The effects and complications expected from different therapies depend on the characteristics of the primary tumour, on the depth of invasion of the oesophageal wall and neighbouring structures, and the metastasis of the cancer. Hence, the correct identification of the stage of the oesophagus carcinoma turns into a crucial task in order to give the adequate therapy to each patient. In order to perform this task, the filter and wrapper approaches to Bayesian classification models presented in Chapter 9 are used.

12.2 The oesophageal cancer domain

12.2.1 The oesophageal cancer network

With the help of two experts in gastrointestinal oncology from the Antoni van Leeuwenhoekhuis, a probabilistic network has been constructed to model the staging of the oesophageal carcinoma van der Gaag et al., 2002. The experts identified the relevant prognostic and diagnostic factors and their corresponding states to be included in the network as nodes. The probabilistic relationships among the variables are represented by edges. These set of influences has variations in strength expressed as conditional probabilities.

Currently, the network is made up of 42 nodes and involves more the 1000 probabilities. Figure 12.1 shows the network structure of the elicited network and the prior marginal probability distributions for each variable.

12.2.2 Patients: cases and variables

The medical staff of the Antoni van Leeuwenhoekhuis collects data of oesophageal carcinoma patients. At this moment, the records of 156 patients diagnosed with oesophageal cancer are collected in a working database.

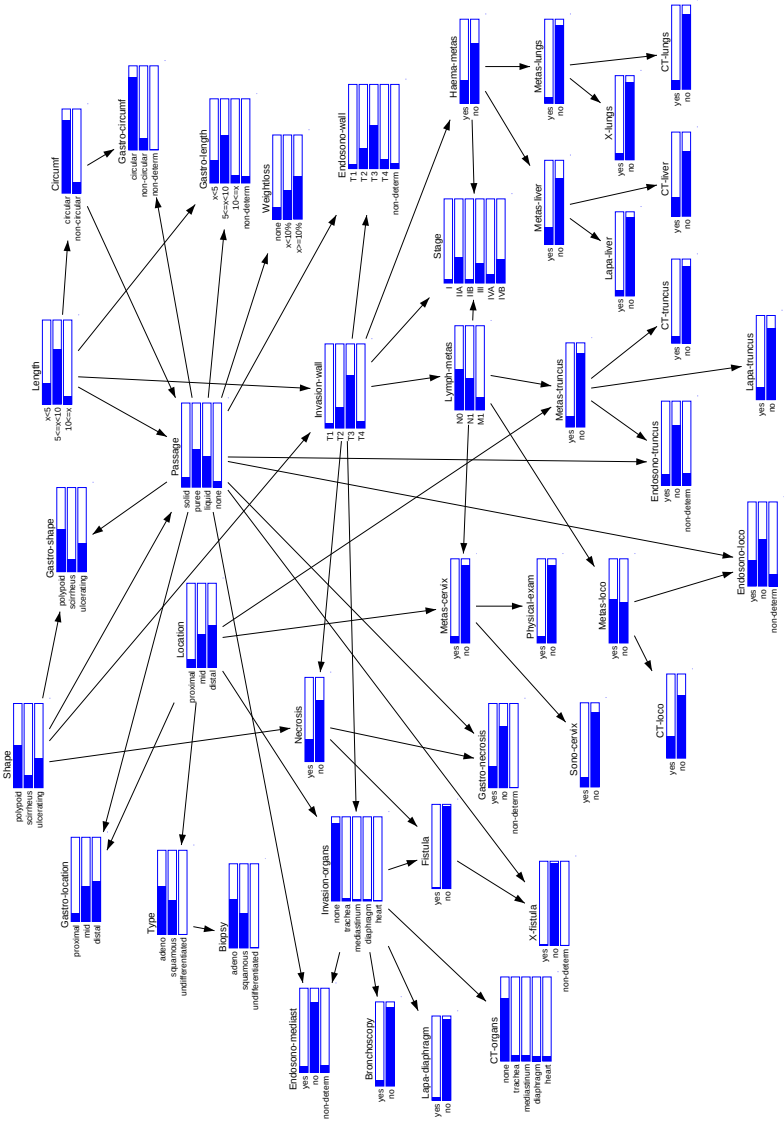


Fig. 12.1. The oesophageal cancer network structure.

Although the Bayesian network which models oesophageal cancer contains 42 variables, these patients' records only include 26 observable variables; 25 are medical test observations and 1 is the stage of the carcinoma (with 6 states).

The observable variables come from the different symptoms and diagnostic tests performed to establish the carcinoma stage. A gastroscopic examination

		n. missing	*ratio*
biopsy		1	0.64%
bronchoscopy		139	89.10%
CT-scan	liver	59	37.82%
	loco-region	45	28.85%
	lungs	43	27.56%
	organs	40	25.64%
	truncus coeliacus	54	34.61%
sonography	neck	124	79.49%
endosonography	loco-region	108	69.23%
	mediastinum	108	69.23%
	wall	108	69.23%
	truncus coeliacus	156	100.0%
gastroscopy	circumference	14	8.97%
	length	0	0.00%
	location	0	0.00%
	shape	9	5.77%
	necrosis	2	1.28%
laparoscopy	liver	137	87.82%
	diaphragm	139	89.10%
	truncus coeliacus	138	88.46%
X-ray	lungs	39	25.00%
	fistula	28	16.02%
physical examination		117	75.00%
interview	passage	2	1.28%
	weight loss	11	7.05%

Table 12.1. Number of values missing for each variable in the patient data.

provides information about the primary tumour such as its length and its location in the oesophagus. The histological or cell type of the tumour is given by a biopsy. In order to determine the presence or absence of haematogenous metastases, tests like the laparoscopic examination of the liver, the CT-scan of the liver and the lungs, and an X-ray of the lungs could be useful. An endosonographic examination provides information about how much the primary tumour has invaded the oesophageal wall.

The stage of the oesophageal carcinoma of each patient is established by the medical staff. This stage could be either I, IIA, IIB, III, IVA or IVB, in order of advanced disease.

Nevertheless, the patient data is a sparse database. For each patient, various diagnostic symptoms and test results are available. The number of data available for each patient ranges between 6 and 21, with an average of 14.8. The instances where the stage of the carcinoma is missing are rejected and removed from the dataset. In Table 12.1, the number of values missing within each observable variable is shown.

		network						
		I	IIA	IIB	III	IVA	IVB	*total*
	I	**2**	0	0	0	0	0	2
	IIA	0	**37**	0	1	0	0	38
data	IIB	0	2	**0**	2	0	0	4
	III	1	17	0	**28**	0	1	47
	IVA	1	1	0	9	**28**	0	39
	IVB	0	1	0	6	4	**15**	26
	total	4	58	0	46	32	16	156

Table 12.2. Confusion matrix established for the oesophageal cancer network with the available patient data.

As a benchmark accuracy to compare the classifiers presented in Section 9, the available data are labelled by the Bayesian network which models the oesophageal carcinoma domain –see Figure 12.1–. Table 12.2 shows the attained confusion matrix. The network correctly label 110 of the 156 patients; hence, the accuracy of the network over the collected data is 70.51%.

12.3 Experimental results

Owing to the intrinsic characteristics of the oesophageal carcinoma domain and the availability of a Bayesian network modelling the problem, several experimental frameworks are performed. In this work, three different experimentations are carried out.

However, the experiments have some characteristics in common. The parameters of the Bayesian classifiers are estimated using the Laplace correction Laplace, 1814 to their maximum likelihood parameter estimations. In order to analyse the filter feature reduction as the α parameter decreases, different values of the α parameter ($\alpha = 0.05$, $\alpha = 0.01$, $\alpha = 0.005$, and $\alpha = 0.001$) are considered. The k of all the kDB classifiers is fixed at $k = 3$.

12.3.1 Working with imputed data

The aim of this experimentation is the identification of the oesophageal carcinoma type with a feature selection reduction process, in order to improve the patients' life expectancy with the adequate therapy for each type of carcinoma. The experimental results focus on the average accuracy and the number of features selected to reach the corresponding accuracy. The accuracy estimation is attained by a *leave-one-out* cross-validation process.

The patient database is sparse, so several processings could be applied. As the Bayesian classifiers presented in Chapter 9 require complete and discrete data, in order to work with the patient data, a simple imputation algorithm is run replacing the missing value with the mode of the variable given the class.

		accuracy	*n. features*	*n. evaluations*
	Naive Bayes	95.51 ± 1.66	25	
	TAN	91.67 ± 2.22	25	
	kDB	91.67 ± 2.22	25	
	Selective NB_{fs}	96.15 ± 1.54	10	
$\alpha = 0.05$	Semi NB_{fs}	93.59 ± 1.96	10	
	TAN_{fs}	96.15 ± 1.54	10	
	$k DB_{fs}$	96.15 ± 1.54	10	
	Selective NB_{fs}	96.79 ± 1.41	9	
$\alpha = 0.01$	Semi NB_{fs}	94.87 ± 1.77	9	
	TAN_{fs}	96.15 ± 1.54	9	
	$k DB_{fs}$	96.79 ± 1.41	9	
	Selective NB_{fs}	96.79 ± 1.41	8	
$\alpha = 0.005$	Semi NB_{fs}	95.51 ± 1.66	8	
	TAN_{fs}	96.79 ± 1.41	8	
	$k DB_{fs}$	96.79 ± 1.41	8	
	Selective NB_{fs}	96.79 ± 1.41	8	
$\alpha = 0.001$	Semi NB_{fs}	96.15 ± 1.54	8	
	TAN_{fs}	96.79 ± 1.41	8	
	$k DB_{fs}$	96.79 ± 1.41	8	
	Selective NB_{ws}	94.87 ± 1.77	5	135
	Semi NB_{ws}	94.23 ± 1.87	5	331
	TAN_{ws}	94.87 ± 1.77	3	127
	$k DB_{ws}$	94.87 ± 1.77	5	249

Table 12.3. Averaged accuracy and standard deviation of leave-one-out cross-validation.

Table 12.3 shows the estimated accuracy (and its corresponding standard deviation) and the number of features of the classifier induced with the whole dataset. In the case of the wrapper approaches, the number of evaluations related to the computational cost of learning the Bayesian classifiers with the whole dataset is also shown.

The Mann-Whitney Mann and Whitney, 1947 statistical test is performed to compare statistically significant differences between the accuracy results. In spite of the non-existence of statistically significant differences, the filter approaches obtain higher results that the corresponding wrapper approaches. Despite the significant feature reduction performed by the wrapper approaches, the accuracy results of these wrapper models are worse than the naive Bayes without feature selection. Nevertheless, all the classifiers attain better accuracy results than the Bayesian network which models the oesophageal carcinoma domain. Even the naive Bayes without feature reduction outperforms the network accuracy.

Table 12.4 shows the variables selected by the Bayesian classifiers. It must be noted that, for the filter approaches, the displayed variables are selected with $\alpha = 0.001$.

		Filter	Sel NB_{ws}	Semi NB_{ws}	TAN_{ws}	$k DB_{ws}$
CT-scan	loco-region	X				
	lungs		X	X		X
	organs	X				
	truncus coeliacus	X				
sonography	neck	X				
endosonography	loco-region	X	X	X	X	X
	mediastinum	X				
	wall	X				
gastroscopy	location		X			X
	shape			X		
laparoscopy	liver	X	X	X	X	X
	truncus coeliacus	X	X	X	X	X

Table 12.4. List of variables included in the Bayesian classifiers when real data are imputed.

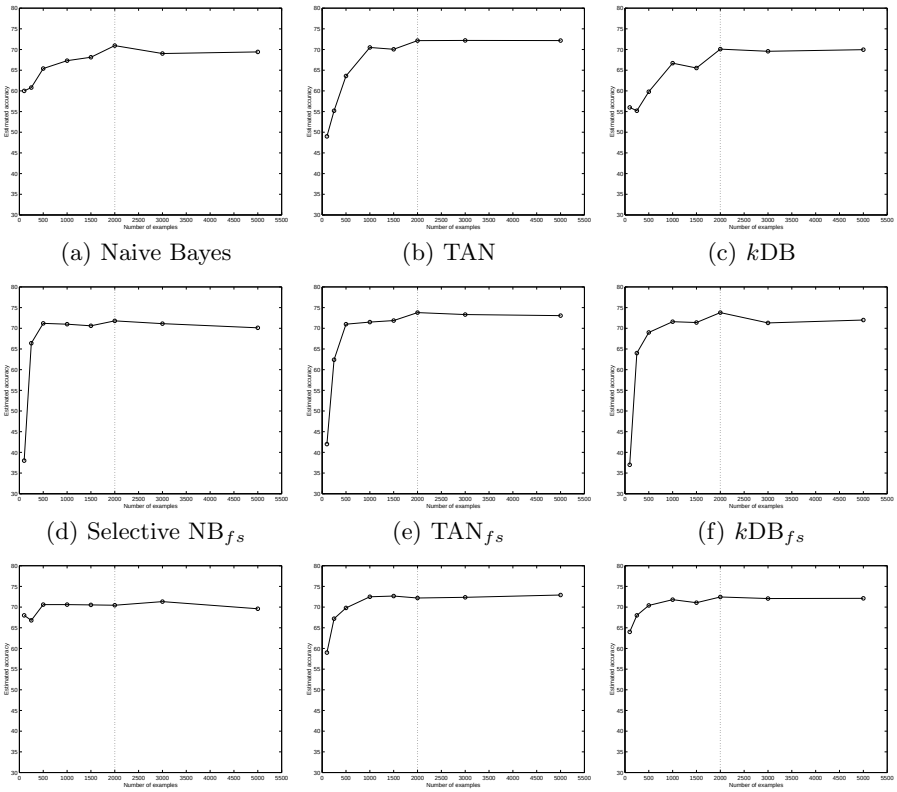
Comparing Table 12.4 with Table 12.1, it must be noted that the three variables selected by all the classifiers (laparoscopy of the liver and the diaphragm and endosonography of the loco-region) have a large missing values ratio. Bearing in mind the high accuracy estimation of the classifiers over the imputed data and comparing it with the accuracy results of the Bayesian network which models the domain, it seems that the imputation algorithm influences the results. Thus, the variables with a high missing value ratio have a larger number of imputed values and it seems that they lead the search process to unreliable classifiers. Hence, a small number of these variables distinguish easily among the class states and make the accuracy questionable. Surprisingly, the wrapper approaches select these variables with a high ratio of imputed values and lead the feature reduction to poor subsets of variables.

12.3.2 Learning with sampled data and testing with real data

Due to the fact that the imputation could influence the accuracy results and the availability of the Bayesian network which models the oesophageal cancer domain, an artificial dataset, sampled from the oesophageal carcinoma Bayesian network, is used for Bayesian classifier induction. The classifiers are later tested the real patient data.

In this experimental framework, a Bayesian classifier is built by means of a complete training set. Since the test set is relatively sparse, an algorithm to perform abduction over the classifier is required to propagate the evidence of each patient record. Then, after abduction, the most probable class value is assigned to the patient record. Therefore, not all the Bayesian classifiers presented in this work could be performed this way. As a way to abduct in the supernodes of the seminaive Bayes classifier is unknown, this classification model is overlooked.

In order to fix the sampled data size, artificial databases of 100, 250, 500, 1000, 1500, 2000, 3000 and 5000 instances containing the 25 variables of the real dataset are simulated independently of the oesophageal Bayesian network.



(a) Naive Bayes (b) TAN (c) kDB

(d) Selective NB$_{fs}$ (e) TAN$_{fs}$ (f) kDB$_{fs}$

(g) Selective NB$_{ws}$ (h) TAN$_{ws}$ (i) kDB$_{ws}$

Fig. 12.2. Accuracy evolution with respect to the number of sampled examples.

Then, the Bayesian classifiers are induced over these sampled databases. Figure 12.2 depicts the evolution of the accuracy estimation average with respect to the database size for each Bayesian classifier.

It must be observed in Figure 12.2 that, in almost the classifiers, a logarithmic curve is obtained –with irregularities– when the sample size is small. This means that a small database is not big enough to obtain reliable and stable classifiers and a large database requires higher computational costs to reach a similar classifier. Moreover, the highest value is attained with 2000 instances. It is maintained and even decreases with larger database size. This fact supports the use of the 2000 sample database for Bayesian classifier induction.

Once the sample size is fixed, the induction of Bayesian classifiers is performed with the artificial database. The induced classifiers are tested over the real patient data as explained before.

		10-*fold cv*	*real data*	*n. features*	*n. evaluations*
	Naive Bayes	70.95 ± 2.61	66.67	25	
	TAN	72.15 ± 4.67	67.31	25	
	kDB	70.10 ± 3.31	62.82	25	
	Selective NB$_{fs}$	71.85 ± 2.89	66.67	17	
$\alpha = 0.05$	TAN$_{fs}$	73.85 ± 3.68	66.67	17	
	kDB$_{fs}$	$73.90 \pm 3.12^{\dagger}$	67.31	17	
	Selective NB$_{fs}$	71.80 ± 2.15	67.31	16	
$\alpha = 0.01$	TAN$_{fs}$	73.80 ± 3.42	66.67	16	
	kDB$_{fs}$	$73.80 \pm 2.38^{\dagger}$	68.59	16	
	Selective NB$_{fs}$	70.85 ± 3.14	67.31	15	
$\alpha = 0.005$	TAN$_{fs}$	$73.65 \pm 1.86^{\dagger}$	67.31	15	
	kDB$_{fs}$	73.30 ± 2.66	69.23	15	
	Selective NB$_{fs}$	71.35 ± 2.64	67.31	15	
$\alpha = 0.001$	TAN$_{fs}$	74.00 ± 4.10	67.31	15	
	kDB$_{fs}$	73.65 ± 3.78	69.23	15	
	Selective NB$_{ws}$	70.45 ± 3.34	50.64	10	220
	TAN$_{ws}$	72.20 ± 2.97	52.56	7	454
	kDB$_{ws}$	72.45 ± 3.58	54.56	7	452

Table 12.5. Averaged accuracy and standard deviation of ten-fold cross-validation, accuracy and number of selected features when training with the sampled dataset and testing with the real dataset.

Table 12.5 shows the accuracy over the real data of the Bayesian classifiers induced with the artificial database, and the number of selected features. In the case of the wrapper approaches, the number of solutions visited during the search process is also displayed. To check the quality of the artificial induction, a 10-fold cross-validation process is performed and Table 12.5 displays the corresponding accuracy and standard deviation.

The Mann-Whitney test Mann and Whitney, 1947 is run to compare the differences in accuracy between the classifiers proposed and the naive Bayes when the 10-fold cross-validation is performed. The symbol † in Table 12.5 denotes statistically significant differences at the 0.05 confidence level with respect to the naive Bayes model.

In the results of the 10-fold cross-validation, the Bayesian classifiers that allow dependence relationships among the predictive variables reach a higher accuracy than the corresponding selective naive Bayes classifiers. Although only the kDB$_{fs}$ model with $\alpha = 0.05$ and $\alpha = 0.01$ and the TAN$_{fs}$ classifier with $\alpha = 0.005$ obtain higher results with statistically significant differences with respect to naive Bayes, the average accuracy of almost all the novel approaches is slightly higher than the corresponding model without feature reduction. In spite of the significant feature reduction, the wrapper approaches do not outperform the corresponding filter approach.

The results of training with an artificial database and testing with real patient data reveals similar behaviour when the 10-fold cross-validation process

		selective NB_{ws}						
		I	IIA	IIB	III	IVA	IVB	*total*
	I	**1**	1	0	0	0	0	2
	IIA	0	**37**	0	0	0	1	38
data	IIB	0	2	**0**	2	0	0	4
	III	0	19	2	**23**	1	2	47
	IVA	0	11	0	23	**4**	1	39
	IVB	0	4	0	8	0	**14**	26
	total	1	74	2	56	5	18	156

Table 12.6. Confusion matrix established from the real patient data for the wrapper selective naive Bayes classifier. The classifier includes ten features.

		selective NB_{fs}						
		I	IIA	IIB	III	IVA	IVB	*total*
	I	**1**	1	0	0	0	0	2
	IIA	0	**38**	0	0	0	0	38
data	IIB	0	2	**0**	2	0	0	4
	III	0	19	2	**21**	3	2	47
	IVA	0	2	0	9	**27**	1	39
	IVB	0	0	0	4	4	**18**	26
	total	1	62	2	36	34	21	156

Table 12.7. Confusion matrix established from the real patient data for the filter selective naive Bayes classifier with $\alpha = 0.001$. The classifier includes fifteen features.

is carried out. However, while the wrapper approaches perform competitively with the filter approaches, in the real patient data the wrapper approaches show a considerably decreased accuracy.

An explanation to this fact is found looking at the confusion matrices of all the classifiers. Even though Table 12.6 and 12.7 display the confusion matrices for the filter selective naive Bayes and the wrapper selective naive Bayes models respectively, this behaviour is observed in all the confusion matrices for filter and wrapper classifiers.

The poor accuracy of the wrapper selective naive Bayes (and the other wrapper models) could be attributed to its relative inability to identify patients with a stage IVA carcinoma; as it classifies almost all these patients with a stage III instead. An oesophageal carcinoma in stage IVA is distinguished from a stage III cancer by the presence of secondary tumours in distant lymph nodes. In order to study whether the lymph nodes of the upper abdomen (distant from the primary tumour) are affected by the cancer, three diagnostic tests are available: a CT-scan, an endosonographic examination and a laparoscopic examination. The laparoscopic examination is the most reliable of the three medical tests. However, it involves a surgical procedure and it is only performed in 18 of the 156 patients of the real dataset.

		Filter	Sel NB_{ws}	TAN_{ws}	kDB_{ws}
CT-scan	liver	X	X	X	X
	loco-region	X	X	X	X
	lungs	X			
	organs				X
	truncus coeliacus	X			
sonography	neck	X			
endosonography	loco-region	X	X	X	X
	mediastinum	X			
	wall	X	X	X	X
	truncus coeliacus	X			
gastroscopy	length	X	X		
	necrosis	X	X	X	
laparoscopy	liver	X			
	diaphragm	X	X		
	truncus coeliacus	X	X	X	X
X-ray	lungs	X	X	X	X
physical examination		X	X		

Table 12.8. List of variables included in the Bayesian classifiers when training with the sampled dataset and testing with the real dataset.

Table 12.8 shows the selected variables for each Bayesian classifier (with $\alpha = 0.01$ for the filter models). It could be observed that while the filter approaches select all the observable variables related to the truncus coeliacus (the results of the CT-scan, the endosonography and the laparoscopy), the wrapper approaches select only one of them. Thus, in the filter approaches the importance of these three variables is revealed, so the most patients with stage IVA cancer are correctly labelled. Nevertheless, the wrapper approaches discover the strong conditional dependence relationships among the three variables and only select the laparoscopic examination. However, the laparoscopic examination is an infrequent surgical procedure, so the wrapper classifiers are not able to identify stage IVA cancer.

It seems that the wrapper approaches are too restrictive and are not able to select enough variables to correctly distinguish between stage III cancer and stage IVA cancer. Since the learning database is a complete dataset and the testing database is sparse, feature selection on the artificial database is useless, or at least inadequate, when definitively tested on the real patient data.

12.3.3 Including knowledge from the real data in the learning process

In order to avoid restrictive classifiers in the oesophageal carcinoma domain, as pointed out in the previous section, knowledge of the real domain should be included in the learning process to improve the accuracy results of the classification task.

When the novel approaches to Bayesian classifiers induction are carried out, knowledge of the patient records is added to the learning process in the following way :

		accuracy	*n. features*	*n. evaluations*
	Naive Bayes	66.67	25	
	TAN	67.31	25	
	kDB	62.82	25	
	Selective NB$_{fs}$	64.74	6	
$\alpha = 0.05$	TAN$_{fs}$	64.74	6	
	kDB$_{fs}$	64.74	6	
	Selective NB$_{fs}$	39.10	1	
$\alpha = 0.01$	TAN$_{fs}$	39.10	1	
	kDB$_{fs}$	39.10	1	
	Selective NB$_{fs}$	39.10	1	
$\alpha = 0.005$	TAN$_{fs}$	39.10	1	
	kDB$_{fs}$	39.10	1	
	Selective NB$_{fs}$	39.10	1	
$\alpha = 0.001$	TAN$_{fs}$	39.10	1	
	kDB$_{fs}$	39.10	1	
	Selective NB$_{ws}$	73.07	10	220
	TAN$_{ws}$	73.07	10	550
	kDB$_{ws}$	73.07	10	550

Table 12.9. Averaged accuracy and standard deviation when knowledge about the domain is used to induce Bayesian classifiers.

- Filter approaches: the $2N_iI(X_i, C)$ attained from the real patient data, where N_i is the number of available instances, have to overcome the corresponding χ^2 value.
- Wrapper approaches: the accuracy of the current solution over the real patient data, calculated using an abduction method, guides the search process.

It must be noted that all classifier parametric learning is performed over the artificial database. In this way, the structural learning is run with the help of the patient records, but the parametric estimation is not biased by the high number of missing values. Once the Bayesian classifier is learned, it is tested over the real patient data.

Table 12.9 shows the accuracy results, the number of selected features and, in the case of wrapper approaches, the number of solutions visited during the search. It must be observed that accuracy increases in the wrapper approaches and decreases in the filter approaches, especially when $\alpha \leq 0.01$. It is remarkable that the filter approaches with $\alpha \leq 0.01$ only select one variable and perform considerably worse than the other approaches. Now the wrapper approaches even obtain the highest accuracy results. However, these results must be treated with caution due to the inclusion of the same dataset in the learning and in the testing processes. The use of the same dataset for learning and testing in classification tasks makes accuracy unstable and unreliable when new instances are tested.

		Filter	Sel NB$_{ws}$	TAN$_{ws}$	kDB$_{ws}$
CT-scan	liver	X	X	X	X
	loco-region	X			
	lungs	X	X	X	X
	organs		X	X	X
	truncus coeliacus	X	X	X	X
sonography	neck		X	X	X
endosonography	loco-region	X	X	X	X
	wall		X	X	X
laparoscopy	liver		X	X	X
	truncus coeliacus		X	X	X
X-ray	lungs	X	X	X	X

Table 12.10. List of variables included in the Bayesian classifiers when knowledge of the domain is used to induce Bayesian classifiers.

Table 12.10 shows the variables selected by the novel approaches, with $\alpha = 0.05$ in the case of the filter approaches. Now the subset of variables included in the classification model are 'important enough' in the real patient dataset. Hence, the variables included have a relatively small number of missing values. Therefore, from the group of sixteen features selected in the former section when the filter approaches are taken into account, only six have 'enough' available data. However, due to the high number of values missing in the laparoscopic examination, the wrapper approaches are able to detect the relevance of the CT-scan to distinguish between stage III cancer and stage IVA cancer.

13

Selection of accurate genes in the DNA microarray domain

For the past years the use of DNA microarrays has grown spectacularly. Instead of one or two genes of an organism, DNA microarrays let thousands of gene expression activation levels be measured and monitored simultaneously in a single experiment Brown and Botsein, 1999. This way, researchers can analyse and study thousand of genes at a time. Furthermore, from the biological point of view, DNA microarrays provide a tool to understand the networks of biomolecular interactions at a global scale.

Technical improvement has given rise to the storage and analysis of microarray data, which can greatly reshape biomedical science. By studying DNA microarrays, the possibility of obtaining answers to old and new biological questions is open. A systematic and computational analysis of microarray databases is a promising way to understand underlying biological processes. Several tools, like classical hypothesis testing, analysis of variance, image processing and analysis and clustering Drăghici, 2003, have been used for this purpose. However, a popular paper, one of the first works on DNA microarrays analysis, Golub et al. (1999) proposes class prediction (or supervised classification) as the task.

This chapter is organised as follows. The next section introduces the central dogma of molecular biology. Section 13.2 explains how DNA microarrays are constructed. Finally, the experimental results over two well-known DNA microarray datasets are presented by the use of Bayesian classifiers.

This chapter is an adaptation of the works of Blanco et al. (2001), Blanco et al. (2004b), Inza et al. (2002b), Inza et al. (2002a), Inza et al. (2004) and Larrañaga et al. (2002).

13.1 Introduction

DNA carries genetic instructions for the biological development of all cellular forms of life. In bacteria and other simple-cell organisms, DNA is distributed

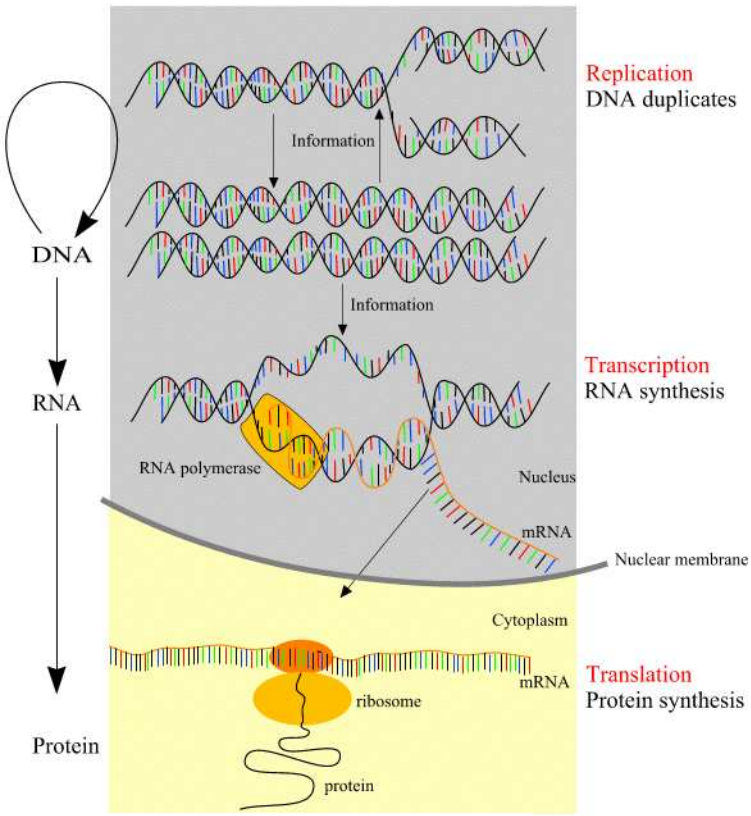


Fig. 13.1. Central dogma of molecular biology.

throughout the cell. But in the complex cells that make up plants, animals and other multicellular organisms, DNA is located in the cell's nucleus.

DNA is composed of two long strands of four bases: adenine (A), thymine (T), cytosine (C) and guanine (G). The bases' strands are joined together by weak hydrogen bonds which produce the double helix shape. Due to the fact that the only pairs allowed are A-T and C-G, the strands are complementary.

As DNA folds to form the chromosomes, DNA is sometimes referred to as the molecule of heredity. The basic unit of heredity is the gene, which is composed of a sequence of nucleotide bases. A gene specifies a protein by means of its base chain. These molecules are polymer chains of amino acids. In order to carry out polymer synthesis, the gene information has to go to the cytoplasm for protein synthesis. Since DNA is degraded out of the nucleus, a molecule is required to store and move the information. This molecule is the messenger RNA (mRNA). The mRNA is a copy of one of the DNA strands, and, this way, the gene information goes out of the nucleus

and is used for protein synthesis. The processes of DNA replication, RNA synthesis and protein synthesis are the 'central dogma of molecular biology'. Figure 13.1 sums up the dogma.

13.2 DNA microarray fabrication

A DNA microarray is usually a substrate (nylon membrane, glass or plastic) on which one deposits single stranded DNA (ssDNA) with various sequences. Depending on the purpose of the array and the technology used, the deposit differs. The array is used to answer a specific question regarding the DNA on the surface. Usually, the microarray surface is washed with a solution containing ssDNA, called target, to perform the question. The idea is that the sequences of the DNA in the solution hybridise to those complementary sequences deposited on the surface of the array. Since the target is labelled with a fluorescent dye or another method, the hybridisation spot can easily be detected and quantified.

When gene expression studies are carried out, the DNA target used to hybridise the array is obtained by reverse transcription from the mRNA extracted from a tissue example. This means that the mRNA transcripted from the DNA is transcripted to reach an identical copy of a DNA strand. The DNA target is fluorescently labelled with a dye and illumination with an adequate light produces an image of features. The intensity of each spot or the average difference between matches and mismatches could be related to the amount of mRNA present in the tissue and, hence, with the number of proteins produced by the gene corresponding to the given feature.

There are two main approaches to microarray fabrication: deposition of DNA fragments and *in situ* synthesis. These approaches differ considerably on the methodology applied to obtain the DNA microarray.

Deposition based fabrication of cDNA microarrays Brown and Botsein, 1999 is largely used due to its flexibility and relatively low economical cost. The DNA is not directly prepared on chip. Robots dip thin pins into solutions containing the desired DNA material and then touch the surface arrays with them. Small quantities of DNA are deposited on the array in the form of spots. Spotted arrays could use small sequences or whole genes. It must be noted that by means of this technique, the relative differences in amount of mRNA of two hybridised tissues (control and sample) are calculated. Therefore, the red and green colours are applied to measure of ratio between the sample and the control. The green spots mean that only the control tissue has been expressed and the red spots denote that only the sample has been expressed. Thus, the yellow spots indicate that both control and sample have been expressed, and finally, the black spots mean that neither the control nor the sample have been expressed. Figure 13.2 displays a scheme of the cDNA technique.

During the microarray fabrication process based on *in situ* synthesis, the samples are photochemically synthesised on the chip. As there is no cloning

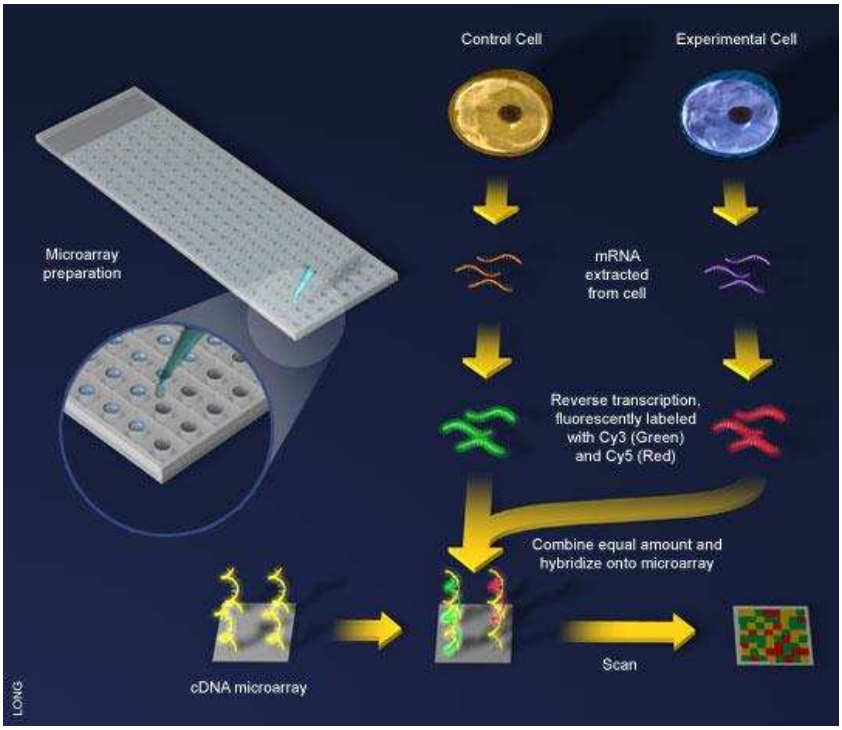


Fig. 13.2. Schematised experimental process of cDNA technique for microarray construction.

nor spotting, *in situ* synthesis is not noisy; this is the main advantage of this approach. There are three approaches to the *in situ* technique. The most popular approach is the Affymetrix process (called this way because of the company which marketed it). This method uses photolithographic masks for each base in a way similar to the technology used to build very large scale integrated circuits in computers. If a sample has a base, the corresponding mask has a hole in which the base must be deposited. Masks construct the sequences base by base. This technique permits the fabrication of very high density arrays, although the length of the DNA sequence constructed is limited. Figure 13.3 shows the process of Affymetrix microarray technique.

Both fabrication processes, cDNA and Affymetrix, provide images where spots represent the gene expression level. As a DNA microarray is an image, methods to turn the images into expression level matrices are required. Classical image processing algorithms are applied at this step. Once the expression level matrix is reached, the analysis of the gene expression levels can be carried out.

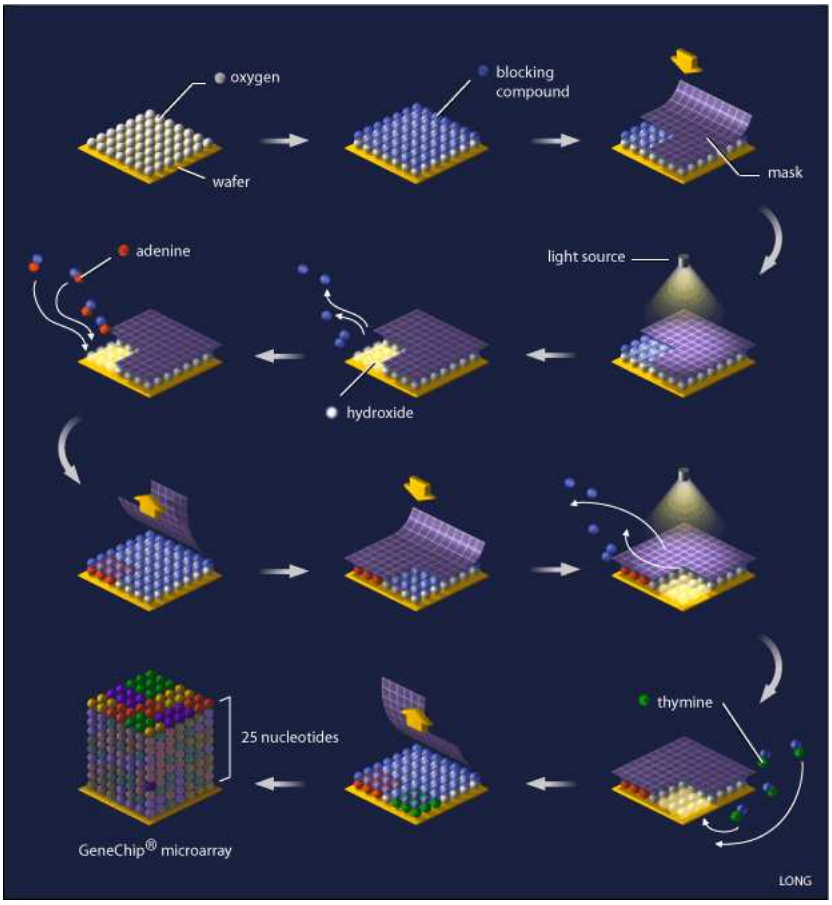


Fig. 13.3. Schematised experimental process of the Affymetrix microarray construction technique.

13.3 Experimental results

Two well-known DNA microarray domains are used to induce and test the Bayesian classification models proposed in Chapter 9:

- The *colon* dataset Alon et al., 1999 has 62 samples of colon epithelial cell-lines. The samples are collected from colon cancer patients. The 'tumour' biopsies are collected from tumours (22 samples) and the 'normal' biopsies were collected from healthy tissues of the colons of the same patient (40 samples). The original dataset contains around 6000 genes, but only 2000 are selected by the authors based on the confidence in their measured expression level. The task is to distinguish between the healthy and the tumour tissues.

- The *leukaemia* dataset Golub et al., 1999 includes 72 cell-lines of leukaemia patients involving 7129 genes. All the tissues come from tumour tissues. The task is to distinguish between two different types of leukaemia: acute myeloid leukaemia-AML (25 samples) and acute lymphoblastic leukaemia-ALL (47 patients).

Due to the dimensions of the databases (thousands of variables and dozens of cases), complex models tend to overfit the dataset. Moreover, inducers which take into account relationships among the variables require a high computational cost. Thus, in this application of the Bayesian classification models, the experimentation is limited to the selective naive Bayes presented in Section 9.3.

Although the *colon* and *leukaemia* datasets are well-known sets used previously in the literature, an exhaustive comparison between the results of this work and the results of the literature is not a fair comparison due to the different methodologies applied. However, competitive results are achieved for both datasets. Since these datasets are popular DNA microarray data, the results for *leukaemia* could be consulted, for instance, in: Bø and Jonassen (2002), Lin and Johnson (2000), Keller et al. (2000), Pérez et al. (2002) and Xing et al. (2001). For the *colon* dataset, the following papers could be seen: Ben-Dor et al. (2000), Bø and Jonassen (2002) and Keller et al. (2000).

The wrapper approach to selective naive Bayes is a time-consuming method. Nevertheless, given the promising accuracy results of the previous section, it is worth studying the wrapper selective naive Bayes more thoroughly.

The approach proposed has been carried out over two well-known biological datasets. In this case, continuous and discrete values are taken into account. For the discrete naive Bayes models, each gene is discretised into two values depending on its corresponding median.

After the image-processing methods, the expression level matrix of the DNA microarray is reached. This expression level matrix has continuous variables to represent the DNA microarray brightly. In order to work with Bayesian classifiers, the continuous values are usually discretised. However, in this experimentation a comparison between discrete and continuous values is performed. For discrete naive Bayes models, each gene is discretised into two values depending on its corresponding median. For continuous variables, how to measure the most probable class should be introduced. Thus, the most a posteriori probable class for a naive Bayes with continuous variables is calculated as follows:

$$c^* = arg\max_c p(C=c)\prod_{i=1}^{n} f_{X_i|C=c}(x_i|c)$$

where $f_{Xi|C=c}(x_i|c)$ represents the density function of the X_i variable given $C = c$. In this work, it is assumed that the previous density conditioned functions follow a normal distribution. That is, for all $i = 1, \ldots, n$ and $c = 0, 1$:

$$f_{X_i|C=c}(x_i|c) \rightsquigarrow \mathcal{N}(x_i; \mu_i^c, (\sigma_i^c)^2)$$

Finally, c^* is computed as:

$$c^* = arg\max_c p(C=c) \prod_{i=1}^{n} \left[\frac{1}{\sqrt{2\pi}\sigma_i^c} e^{-\frac{1}{2}\left(\frac{x_i-\mu_i^c}{\sigma_i^c}\right)^2} \right]$$

Aside from the wrapper greedy method presented in Section 9.3, the estimation of distribution algorithms –see Chapter 3 and Section 5.4– could be a helpful search method.

Usually, the starting population of the EDAs is randomly generated. However, taking the wide dimension of DNA microarrays into account, an appropriate initialisation of the search could save a great deal of computation time Aha and Bankert, 1994. The search initialisation is based on the simulation of a probability distribution for each gene. Four different initialisations are compared, three of them based on the results of the wrapper greedy algorithm and naive Bayes.

Based on the feature subset obtained by the greedy wrapper algorithm, three initialisations for EDAs are proposed:

- init-A
  This initialisation assigns the same probability to all the features of the dataset. This probability is calculated taking the number of genes selected by the greedy wrapper algorithm into account. The genes of the individuals of the first population of EDAs have the same probability of being chosen.
- init-B
  In this case, all the genes of the dataset are handled in the same way. The probability assigned to each gene G_i is determined by means of the estimated accuracy of the classification model built only with the class variable and G_i (this means that the rest of genes or features are rejected and the assigned probability is proportional to $Acc(G_i)$). With init-B, the genes with a higher estimated accuracy appear more frequently in the individuals of the first population of EDAs.
- init-C
  Finally, init-C differentiates between the features selected by the greedy wrapper method and the non-selected features. The selected features are assigned with a probability proportional to the improvement of the estimated accuracy when added to the classification model. That is, if $\mathcal{M}_t$ is the classification model with t features and the feature G_i is added to the classifier, then, G_i is assigned with a probability proportional to $Acc(\mathcal{M}_t \cup G_i) - Acc(\mathcal{M}_t)$.
  The non-selected features have a probability of being chosen in the first population of EDAs proportional to $1 - Acc(\mathcal{M})$, where $\mathcal{M}$ is the classification model built with the features selected by the greedy wrapper approach.

		NB		Sel. NB$_{ws}$	
		acc. ± dev.	genes	acc. ± dev.	genes
colon	discrete	70.97 ± 5.81	2000	91.93 ± 3.49	5
	continuous	53.23 ± 6.39	2000	95.83 ± 2.56	3
leukaemia	discrete	63.89 ± 5.70	7129	98.61 ± 1.39	6
	continuous	84.72 ± 4.27	7129	87.09 ± 3.98	2

Table 13.1. Average accuracy and number of selected genes for the *colon* and *leukaemia* domains with continuous and discrete data.

	colon				*leukaemia*			
	discrete		continuous		discrete		continuous	
	acc.	genes	acc.	genes	acc.	genes	acc.	genes
init-0	67.74	985	74.19	1069	45.8	3402	76.39	3587
init-A	95.16	13	98.39	6	100	8	100	10
init-B	95.16	13	98.39	10	98.61	15	100	11
init-C	91.93	5	95.16	3	98.61	6	98.61	4

Table 13.2. Best accuracy estimated by the EDAs and corresponding number of genes for the *colon* and *leukaemia* domains with continuous and discrete data.

It must be noted that, for the three initialisation methods, the expected number of genes selected in each individual of the first population of EDAs is the number of features finally selected by the greedy wrapper method.

Apart from these initialisations, the init-0 is not dependent on the feature subset obtained by the greedy wrapper. In this initialisation, each feature or gene is chosen with a probability of 0.5. This means that, in the individuals of the first population of EDAs, the expected number of selected genes is half the total number of genes.

In the EDA approach proposed, the population size is fixed to 100 individuals, and 50 individuals are selected in order to learn the probability distribution model. The search stops when the sum of the scoring function of the previous population is equal to the sum of the scoring function of the current population.

Table 13.1 shows the results of the *leave-one-out* validation process with all the features of the problem domain and with the features selected by the wrapper selective naive Bayes presented in Section 9.3. The results support the fact that not all the features are relevant in order to learn the classification model or the existence of redundant features. These results follow the findings of Golub et al. (1999) and Xing et al. (2001), relating the low number of features needed to improve the accuracy of the whole feature set.

For each dataset and initialisation method, ten EDA independent runs have been executed. Table 13.2 shows the estimated accuracy of naive Bayes and the number of features selected for the best run of each initialisation method. Table 13.3 shows the estimated average accuracy, the average number

			acc. ± dev.	genes ± dev.	gener. ± dev.
colon	discrete	init-0	64.5 ± 0.2	987 ± 39.1	29.0 ± 6.9
		init-A	91.9 ± 0.1	11.9 ± 4.1	13.0 ± 4.0
		init-B	91.2 ± 0.2	11.8 ± 3.2	11.8 ± 3.2
		init-C	90.9 ± 0.1	6.3 ± 1.6	3.9 ± 1.6
	continuous	init-0	64.9 ± 10.5	1035 ± 52.4	19.14 ± 8.7
		init-A	95.0 ± 2.3	7.1 ± 2.1	15.2 ± 4.6
		init-B	94.7 ± 2.9	7.2 ± 2.4	12.7 ± 6.9
		init-C	93.4 ± 1.6	6.0 ± 1.9	12.8 ± 5.0
leukaemia	discrete	init-0	44.0 ± 0.1	3476 ± 57.0	18.2 ± 6.7
		init-A	97.2 ± 0.1	14.6 ± 3.6	14.2 ± 4.2
		init-B	96.9 ± 0.1	14.8 ± 3.6	12.9 ± 4.7
		init-C	98.6 ± 0.0	8.1 ± 1.8	3.3 ± 1.2
	continuous	init-0	75.9 ± 0.8	3561 ± 35.9	9.3 ± 1.5
		init-A	98.8 ± 1.8	11.0 ± 3.6	18.1 ± 5.7
		init-B	98.8 ± 1.5	11.8 ± 3.2	16.3 ± 3.6
		init-C	96.3 ± 1.1	3.7 ± 1.1	5.9 ± 5.0

Table 13.3. Average estimated accuracy, number of features and average generation where the best solution of the run appears for the *colon* and *leukaemia* domains with discrete and continuous genes.

of selected features for the ten executions of each initialisation method and the average generation where the best solution of the execution is shown.

Although EDAs in the continuous model do not report a significant accuracy improvement with respect to the greedy wrapper in the *colon* dataset, the opposite behaviour, obtaining a significant accuracy improvement by EDA techniques, is shown in the *leukaemia* domain. However, the use of an extremely low number of features is not recommended in previous works Golub et al., 1999 because the use of a very small number of genes (Golub et al. (1999) fixes 10) may produce a classification model which depends too heavily on any gene, producing spuriously high prediction strengths.

Previous works in this type of problems Golub et al., 1999; Xing et al., 2001 warn us about their somewhat arbitrary choice in the number of genes finally selected. Thus, stating the problem as an optimisation task and by means of a population based algorithm waiting for convergence, non-arbitrary choices in the search space are taken. Moreover, this way obtains results competitive with the works previously cited.

The Kruskall-Wallis Kruskal and Wallis, 1952 test is carried out to compare the accuracy results, the number of genes and the number of generations required to stop the A, B, and C initialisations. In addition, the Mann-Whitney Mann and Whitney, 1947 test is carried out to check the differences between discrete and continuous approaches.

Table 13.4 reports the p-values when comparing the accuracy results, the number of genes and the number of generations required to stop the three ini-

		acc.	genes	gener.
colon	discrete	$p = 0.440$	$p = 0.002$	$p < 0.001$
	continuous	$p = 0.232$	$p = 0.446$	$p = 0.187$
leukaemia	discrete	$p = 0.004$	$p = 0.001$	$p < 0.001$
	continuous	$p = 0.003$	$p < 0.001$	$p < 0.001$

Table 13.4. p-values when comparing A, B, and C initialisations.

		discrete vs. continuous		
		accuracy	genes	gener.
colon	init-0	$p = 0.47$	$p = 0.042$	$p = 0.174$
	init-A	$p = 0.007$	$p = 0.009$	$p = 0.353$
	init-B	$p = 0.043$	$p = 0.004$	$p = 0.631$
	init-C	$p = 0.001$	$p = 0.631$	$p < 0.001$
leukaemia	init-0	$p = 0.017$	$p = 0.017$	$p = 0.067$
	init-A	$p = 0.063$	$p = 0.063$	$p = 0.075$
	init-B	$p = 0.089$	$p = 0.075$	$p = 0.063$
	init-C	$p < 0.001$	$p < 0.001$	$p = 0.218$

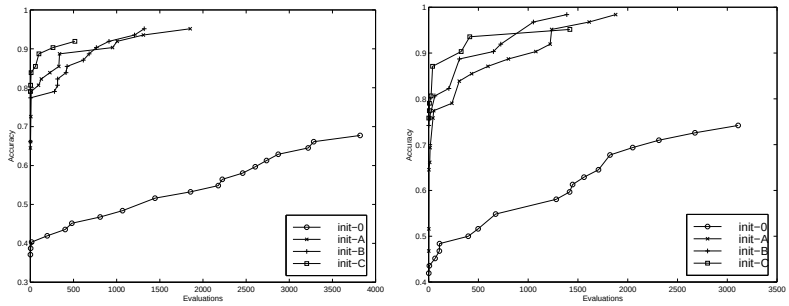
Table 13.5. p-values when comparing discrete versus continuous models.

tialisations proposed, which indicate the probability that one initialisation is better than the others, where a p-value of 0.05 indicates that the initialisations compared are different with a probability of 95%.

In the *colon* database, only significant differences ($p < 0.05$) are obtained in the discrete model in relation to the number of features and the number of generations needed for convergence. In the *leukaemia* database, the test shows that the differences in all criteria with respect to three initialisations are statistically significant in both ways of making the classification: when the dataset is discrete and when it is continuous.

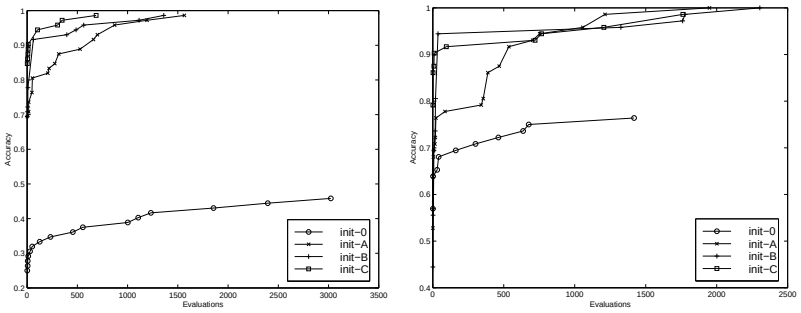
Table 13.5 shows the results obtained when applying the Mann-Whitney Mann and Whitney, 1947 test in order to compare the behaviour between the discrete and continuous selective naive Bayes models.

In the *colon* database, statistically significant differences are found in relation to the accuracy of the model for the initialisations init-A, init-B, and init-C, obtaining the best results in the case of continuous selective naive Bayes. With respect to the number of features selected by the EDA, the continuous selective naive Bayes model needs significantly more features than its corresponding discrete model in init-A and init-B initialisations. Finally, regarding the number of generations needed until convergence is reached, the differences are statistically significant for initialisation init-C, where the discrete model needs a larger number of generations.



(a) discrete (b) continuous

Fig. 13.4. The evolution of the best accuracy found in *colon* dataset: (a) discrete, (b) continuous.



(a) discrete (b) continuous

Fig. 13.5. The evolution of the best accuracy found in *leukaemia* dataset: (a) discrete, (b) continuous.

In the *leukaemia* database, the differences are only statistically significant in the case of initialisation init-C with respect to the accuracy of the model – better results for the discrete selective naive Bayes– and the number of features –fewer features for the continuous classifier.

Figures 13.4 and 13.5 show a typical run of the evolution of the best estimated accuracy found in the search process. The evolution of the *colon* dataset is depicted in Figure 13.4. The evolution of the *leukaemia* dataset is drawn in Figure 13.5. These figures display how the initialisation is used to guide the search in the first steps of the process. It can be seen in all cases that init-0 obtains poor solutions at the first generation. In fact, the starting generation of init-0 obtains worse individuals (in terms of accuracy) than init-A, init-B, and init-C.

14

Conclusions and future work

14.1 Conclusions

The Bayesian classification models presented in Chapter 9 are applied to several biomedical domains with different purposes.

In the cirrhotic patients domain, in order to predict whether a patient would survive at least six months after TIPS placement, a classification task is performed over a dataset collected by the Clínica Universitaria de Navarra. Coupled with an improvement in accuracy, stable and reliable Bayesian classifiers are desirable to satisfy the medical staff. The Bayesian classifiers obtained reach these objectives by means of feature reduction where the 'subjective' medical variables, i.e. those whose value depend on the point of view of the physicians, are rejected. Furthermore, the feature reduction increases the patient well-being due to a decrease in the number of invasive and painful medical techniques.

The oesophageal carcinoma domain is a challenge due to the dataset characteristics. To distinguish between the six stages of the oesophageal cancer is the classification task performed. A very sparse dataset collected from the Netherlands Cancer Institute contains a relatively small number of cases to identify six class states. Moreover, the distribution of the class states is not uniform and two class states gather a high number of patients. Several approaches are performed to face this difficulty. As the imputation method applied seems to make the accuracy results unreliable and the availability of a Bayesian network which models the domain, simulated data are used to learn the Bayesian classifiers which are tested with the real albeit sparse dataset. Although competitive results are obtained by means of this combination of simulated data and real data, variables with a high number of missing values are selected. Thus, in order to overcome the selection of variables with a high number of missing values, knowledge of the real dataset is included in the learning process. This final approach must be taken with caution due to the use of the same dataset in the induction of Bayesian classifiers and in the testing stage.

Finally, an application in the DNA microarray domains is presented. Two well-known DNA microarray datasets are presented to induce the Bayesian classifiers. Nevertheless, the small number of cases (less than 100) in relation to the number of genes (thousands) makes the applied classifier unfeasible when taking into account the relationships among the domain variables. Then, only wrapper approaches to selective naive Bayes are run, where the selective naive Bayes is attained by means of the EDAs. Several initialisations to obtain the first generation of EDAs are proposed. The attained accuracy results and the number of selected variables are supported by previous results from the literature Golub et al., 1999; Xing et al., 2001, and mark the importance of proper initialisations to obtain good individuals.

14.2 Future work

The biomedical domains are a challenge to the machine learning field due to their intrinsic characteristics. Taking into account the difficulties of the three biomedical problems presented, improvements and future work lines emerge:

- Oesophageal carcinoma. The applied imputation technique seems to influence the accuracy results. An intensive and exhaustive experimentation with several imputation methods is a future work line. Moreover, another possible work line is the addition of knowledge of the real data in such a way that the final accuracy results become reliable.
- DNA microarrays. The experimentation over the two datasets could be extended to the filter Bayesian classifiers presented in Chapter 9. In addition, after a pre-processing step to select only a subset of genes, the experimentation could be performed with the filter and wrapper approaches of all the Bayesian classification models.

Additionally, the wrapper approaches to induce Bayesian classifiers by means of EDAs (pointed out as future work in Chapter 10) could be analysed with these medical domains.

Bibliography

Abramson, B., Brown, J. M., Edwards, W., Murphy, A., and Winkler, R. L. (1996). Hailfinder: A Bayesian system for forecasting severe weather. *International Journal of Forecasting*, 12:57–71.

Acid, S. (1999). *Métodos de Aprendizaje de Redes de Creencia. Aplicación a la Clasificación*. PhD thesis, Universidad de Granada. In Spanish.

Acid, S. and de Campos, L. (1996). BENEDICT: an algorithm for learning probabilistic belief networks. In *Proceedings of the International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems*, pages 978–984.

Ackely, D., Hinton, G., and Sejnowski, T. (1985). A learning algorithms for Boltzmann machines. *Cognitive Science*, 9:147–169.

Adams, I., Chan, M., and Clifford, P. (1986). Computer-aided diagnosis of acute abdominal pain: a multicentre study. *British Medical Journal*, 293:800–804.

Adams, N. and Hand, D. (1999). Comparing classifiers when the misallocation costs are uncertain. *Pattern Recognition*, 32(7):1139–1147.

Aha, D. and Bankert, R. (1994). Feature selection for case-based classification of cloud types: An empirical comparison. In *Proceedings of the AAAI-94 Workshop on Case-Based Reasoning*, pages 106–112.

Aha, D., Kibler, D., and Albert, M. (1991). Instance-based learning algorithms. *Machine Learning*, 6:37–66.

Akaike, H. (1974). New look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723.

Aliferis, C., Tsamardinos, I., and Statnikov, A. (2003). HITON, a novel Markov blanket algorithm for optimal variable selection. In *Proceedings of the 2003 American Medical Informatics Association*, pages 21–25.

Alon, U., Barkai, N., Notterdam, D., Gish, K., Ybarra, S., Mack, D., and Levine, A. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. In *Proceedings of the National Academy of Sciences*, volume 96, pages 6745–6750.

Anderson, S., Madigan, D., and Perlman, M. (1997). A characterization of Markov equivalence classes for acyclic digraphs. *Annals of Statistics*, 25:505–541.

Bailey, N. (1964). Probability methods of diagnosis based on small samples. *Mathematics and Computer Science in Biology and Medicine*, pages 103–107.

Baluja, S. (1994). Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical Report CMU-CS-94-163, Carnegie Mellon University.

Baluja, S. and Davies, S. (1997). Combining multiple optimization runs with optimal dependency tress. Technical Report CMU-CS-97-157, Carnegie Melon University.

Bard, J. and Feo, T. (1989). Operations sequencing in discrete parts manufacturing. *Management Science*, 35:249–255.

Bard, J. and Feo, T. (1991). An algorithm for the manufacturing equipment selection problem. *IIE Transactions*, 23:83–92.

Beinlinch, I., Suermondt, H., Chavez, R., and Cooper, G. (1989). The ALARM monitoring system: a case study with two probabilistic inference techniques for belief networks. In *Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine*, pages 247–257.

Ben-Dor, A., Bruhn, L., Friedman, N., Nachman, I., Schummer, M., and Yakhini, Z. (2000). Tissue classification with gene expression profiles. *Journal of Computational Biology*, 7(3–4):559–584.

Bergus, G. (1993). When is a test positive? The use of decision analysis to optimize test interpretation. *Family Medicine*, 25(10):656–660.

Binder, J., Koller, D., Russell, S., and Kanazawa, K. (1997). Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2–3):213–244.

Blake, C. and Merz, C. (1998). UCI repository of machine learning databases. http://www.ics.uci.edu/∼mlearn/MLRepository.html.

Blanco, R., Inza, I., and Larrañaga, P. (2002). Floating search methods in learning Bayesian networks. In *Proceedings of the 1st European Workshop on Probabilistic Graphical Models*, pages 9–16.

Blanco, R., Inza, I., and Larrañaga, P. (2003). Learning Bayesian networks in the space of structures by estimation of distribution algorithms. *International Journal of Intelligent Systems*, 18:205–220.

Blanco, R., Inza, I., and Larrañaga, P. (2004a). Learning Bayesian networks by floating search methods. In *Advances in Bayesian Networks*, pages 181–200. Springer-Verlag.

Blanco, R., Inza, I., Merino, M., Quiroga, J., and Larrañaga, P. (2005). Feature selection in Bayesian classifiers for the prognosis of survival of cirrhotic patients treated with TIPS. *Journal of Biomedical Informatics*. Accepted for publication.

Blanco, R., Larrañaga, P., Inza, I., and Sierra, B. (2001). Selection of highly accurate genes for cancer classification by estimation of distribution algorithms. In *Proceedings of the Workshop 'Bayesian Models in Medicine' held within AIME 2001*, pages 29–34.

Blanco, R., Larrañaga, P., Inza, I., and Sierra, B. (2004b). Gene selection for cancer classification using wrapper approaches. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(8):1373–1390.

Blanco, R., van der Gaag, L., Inza, I., and Larrañaga, P. (2004c). Selective classifiers can be too restrictive: a case-study in oesophageal cancer. In *Biological and Medical Data Analysis. 5st International Symposium on Biological and Medical Data Analysis*, volume 3337 of *Lecture Notes in Computer Science*, pages 212–223.

Blum, A. and Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1–2):245–271.

Bø, T. and Jonassen, I. (2002). New features subset selection procedures for classification of expression profiles. *Genome Biology*, 3(4).

Borgelt, C. and Kruse, R. (2001). An empirical investigation of the K2 metric. In *Proceedings of the 6th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 240–251.

Bornman, P., Krige, J., and Terblanche, J. (1994). Management of oesophageal varices. *Lancet*, 343:1079–1084.

Bouckaert, R. (1994a). Properties of Bayesian belief network learning algorithm. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 102–109.

Bouckaert, R. (1994b). A stratified simulation scheme for inference in Bayesian belief networks. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 110–117.

Bouckaert, R. (1995). *Bayesian Belief Networks: From Construction to Inference*. PhD thesis, University of Utrecht.

Boyle, J., Greig, W., Franklin, D., Harder, R., Buchanan, W., and McGirr, E. (1966). Construction of a model for computer-assisted diagnosis: application to the problem of non-toxic goitre. *Quarterly Journal of Medicine*, 35:565–588.

Bradley, A. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Wadsworth and Brooks.

Brier, G. (1950). Verification of forecasts expressed in terms of probabilities. *Monthly Weather Review*, 78:1–3.

Brown, P. and Botsein, D. (1999). Exploring the new world of genome with DNA microarrays. *Nature Biotechnology*, 14:1675–1680.

Brunk, H., Thomas, D., Elashoff, R., and Zippin, C. (1975). Computer aided prognosis. In *Perspective in Miometrics*.

Buntine, W. (1991). Theory refinement in Bayesian networks. In *Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence*, pages 52–60.

Buntine, W. (1996). A guide to the literature on learning probabilistic networks from data. *IEEE Transactions on Knowledge and Data Engineering*, 8(2):195–210.

Cantú-Paz, E. (2002). Feature subset selection by estimation of distribution algorithms. In *Proceedings of 7th the Genetic and Evolutionary Computation Conference*, pages 303–310.

Castillo, E., Gutiérrez, J., and Hadi, A. (1997). *Expert Systems and Probabilistic Network Models*. Springer-Verlag.

Cattlet, J. (1991). On changing continuous attributes into ordered discrete attributes. In *Proceedings of the European Working Session on Learning*, pages 164–178.

Cerquides, J. (1999). Applying general Bayesian techniques to improve TAN induction. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*.

Cerquides, J. and López de Mántaras, R. (2003). Tractable Bayesian learning of tree augmented naive Bayes classifier. In *Proceedings of the 20th International Conference on Machine Learning*, pages 75–82.

Cestnik, B., Kononenko, I., and Bratko, I. (1987). ASSISTANT-86: A knowledge elicitation tool for sophisticated users. In *Progress in Machine Learning*, pages 31–45. Sigma Press.

Chalasani, N., Clark, W., Martin, L., Kamean, J., Khan, M., Patel, N., and Boyer, T. (2000). Determinants of mortality in patients with advanced cirrhosis after transjugular intrahepatic portosystemic shunting. *Gastroenterology*, 118:138–144.

Chavez, R. and Cooper, G. (1990). A randomized approximation algorithm for probabilistic inference on Bayesian belief networks. *Networks*, 20(5):661–685.

Chen, X. (2003). An improved branch and bound algorithm for feature selection. *Pattern Recognition Letters*, 24(12):1925–1933.

Cheng, J., Greiner, R., Kelly, J., Bell, D., and Liu, W. (2002). Learning Bayesian networks from data: an information-theory based approach. *Artificial Intelligence*, 137(1–2):43–90.

Chickering, D. (1995). A transformational characterization of equivalent Bayesian networks. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 87–98.

Chickering, D. (1996). Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics V*, pages 121–130.

Chickering, D. (2002). Learning equivalence classes of Bayesian network structures. *Journal of Machine Learning Research*, 2:445–498.

Chickering, D., Geiger, D., and Heckerman, D. (1994). Learning Bayesian networks is NP-hard. Technical Report MSR-TR-94-17, Microsoft Research, Advanced Technology Division, Microsoft Corporation.

Chickering, D., Geiger, D., and Heckerman, D. (1995). Learning Bayesian networks: Search methods and experimental results. In *Preliminary Papers of the 5th International Workshop on Artificial Intelligence and Statistics*, pages 112–128.

Chickering, D., Meek, C., and Heckerman, D. (2003). Large-sample learning of Bayesian networks is NP-hard. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, pages 124–133.

Chow, C. and Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467.

Clark, P. and Nibblet, T. (1989). The CN2 induction algorithm. *Machine Learning*, 3(4):261–283.

Cohen, W. (1995). Fast effective rule induction. In *Proceedings of the 12th International Conference in Machine Learning*, pages 115–123.

Conn, H. (1981). A peek at the Child–Turcotte classification. *Hepatology*, 1:1–7.

Coomans, D., Broeckaert, I., Jonckheer, M., and Massart, D. (1983). Comparison of multivariate discrimination techniques for clinical data application to the thyroid functional state. *Methods of Information in Medicine*, 22:93–101.

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

Cornfield, J., Dunn, R., Batchlor, C., and Pipberger, H. (1973). Multigroup diagnosis of electrocardiograms. *Computers and Biomedical Research*, 6.

Cost, S. and Salzberg, S. (1993). A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10(1):57–78.

Cotta, C. and Muruzábal, J. (2002). Towards more efficient evolutionary induction of Bayesian networks. In *Parallel Problem Solving From Nature VII*, volume 2439 of *Lecture Notes in Computer Science*, pages 730–739. Springer-Verlag.

Cowell, R. (2001). Conditions under which conditional independence and scoring methods lead to identical selection of Bayesian network models. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 91–97.

Crichton, N., Hinde, J., and Marchini, J. (1998). Models for diagnosing chest pain: is CART helpful? *Statistics in Medicine*, 16(7):717–727.

Croft, D. and Machol, R. (1987). Mathematical models in medical diagnosis. *Annals of Biomedical Engineering*, 2:69–89.

Dagum, P. and Horvitz, E. (1993). A Bayesian analysis of simulation algorithms for inference in belief networks. *Networks*, 23(5):499–516.

Dasgupta, S. (1999). Learning polytrees. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 131–141.

Dawid, A. (1979). Conditional independence in statistical theory. *Journal of the Royal Statistics Society, Series B*, 41:1–31.

De Bonet, J., Isbell, C., and Viola, P. (1997). MIMIC: Finding optima by estimating probability densities. In *Advances in Neural Information Processing Systems*, volume 9. MIT Press.

de Campos, L. (1998). Automatic learning of graphical models. I: Basic methods. In *Probabilistic Expert System*, pages 113–140. Ediciones de la Universidad de Castilla-La Mancha. In Spanish.

de Campos, L., Fernández-Luna, J., Gámez, J., and Puerta, J. (2002). Ant colony optimization for learning Bayesian networks. *International Journal on Artificial Reasoning*, 31(3):109–136.

de Campos, L. and Huete, J. (2000). A new approach for learning belief networks using independence criteria. *International Journal of Approximate Reasoning*, 24(1):11–37.

de Campos, L. and Puerta, J. (2001). Stochastic local algorithms for learning belief networks: searching in the space of the orderings. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 6th European Conference*, volume 2143 of *Lecture Notes in Computer Science*, pages 228–239. Springer-Verlag.

de Dombal, F. (1991). The diagnosis of acute abdominal pain with computer assistance: worldwide perspective. *Annals of Surgery*, 45:273–277.

Debuse, J. and Rayward-Smith, V. (1999). Feature subset selection within a simulated annealing algorithm. *Journal of Intelligent Information Systems*, 9(1):57–81.

DeGroot, M. (1970). *Optimal Statistical Decisions*. McGraw-Hill.

Delorme, X., Gandibleus, X., and Rodriguez, J. (2004). GRASP for set packing problems. *European Journal of Operational Research*, 153(3):564–580.

Doak, J. (1992). An evaluation of feature selection methods and their application to computer security. Technical Report CSE-92-18, University of California at Davis.

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

Dorigo, M., Maniezzo, V., and Colorni, A. (1996). The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 26:29–41.

Drăghici, S. (2003). *Data Analysis Tools for DNA Microarrays*. Chapman and Hall.

duBolay, G., Teather, D., Harling, D., and Clark, G. (1977). Improvements in computer-assisted diagnosis of cerebral tumours. *British Journal of Radiology*, 50:840–854.

Duda, R. and Hart, P. (1973). *Pattern Classification and Scene Analysis*. Jon Wiley and Sons.

Duda, R., Hart, P., and Stork, D. (2001). *Pattern Classification*. Jon Wiley and Sons.

Edwards, F. and Davies, R. (1984). Use of Bayesian algorithm in the computer-assisted diagnosis of appendicitis. *Surgery in Gynaecology and Obstetrics*, 158:219–222.

Efron, B. (1983). Estimating the error rate of a prediction rule: improvement on cross-validation. *Journal of the American Statistical Association*, 78(282):316–330.

Efron, B. and Tibshirani, R. (1993). *An Introduction to the Bootstrap*. Monograph on Statistics and Applied Probability. Chapman and Hall.

Egan, J. (1975). *Signal Detection: Theory and ROC Analysis*. Series in Cognition and Perception. Academic Press.

Elvira Consortium (2002). Elvira: an environment for probabilistic graphical models. In *Proceedings of the 1st European Workshop on Probabilistic Graphical Models*, pages 222–230.

Etxeberria, R. and Larrañaga, P. (1999). Optimization with Bayesian networks. In *Proceedings of the 2nd Symposium on Artificial Intelligence*, pages 332–339.

Fagiuoli, E. and Zaffalon, M. (2000). Tree-augmented naïve credal classifiers. In *Proceedings of the 8th Information Processing and Management of Uncertainty in Knowledge-Based Systems Conference*, pages 1320–1327.

Fawcett, T. (2004). ROC graphs: notes and practical considerations for researchers. http://www.hpl.hp.com/personal/Tom_Fawcett/papers/ROC101.pdf.

Feo, T. and Bard, J. (1989). Flight scheduling and maintenance base planning. *Management Science*, 35:1415–1432.

Feo, T. and Resende, M. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8:67–71.

Feo, T. and Resende, M. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133.

Feo, T., Resende, M., and Smith, S. (1989). A greedy randomized adaptive search procedure for the maximum independent set. Technical report, AT&T Bell Laboratories.

Fisher, R. (1936). The use of multiple measurements. *Annals of Eugenics*, 7:179–188.

Fleurent, C. and Glover, F. (1999). Improved constructive multi start strategies for the quadratic assignment problem using adaptive memory. *INFORMS Journal on Computing*, 11(2):198–203.

Fox, J., Barber, D., and Bardhan, K. (1980). A quantitative comparison with rule based diagnostic inference. *Methods of Information in Medicine*, 19:210–215.

Friedman, N. (1997). Learning belief networks in presence of missing values and hidden variables. In *Proceedings of the 14th International Conference on Machine Learning*, pages 125–133.

Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, 29(2):131–164.

Friedman, N. and Goldszmidt, M. (1996). Building classifiers using Bayesian networks. In *Proceedings of the 13th National Conference on Machine Learning*, pages 1277–1284.

Friedman, N., Goldszmidt, M., and Lee, T. (1998). Bayesian network classification with continuous attributes: getting the best of both discretization and parametric fitting. In *Proceedings of the 15th National Conference on Machine Learning*.

Friedman, N., Goldszmidt, M., and Wyner, A. (1999). On the application of the bootstrap for computing confidence measures on features of induced Bayesian

networks. In *Proceedings of the 7th International Workshop on Artificial Intelligence and Statistics*.

Friedman, N. and Koller, D. (2003). Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1-2):95–125.

Fryback, D. (1978). Bayes' theorem and conditional non independence of data in medical diagnosis. *Computers and Biomedical Research*, 11:423–434.

Fung, R. and Chang, K. (1990). Weighting and integrating evidence for stochastic simulation in Bayesian networks. In *Proceedings of the 5th Conference on Uncertainty in Artificial Intelligence*, pages 209–220.

Fung, R. and Del Favero, B. (1994). Backward simulation in Bayesian networks. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 227–234.

Gammerman, A. and Thatcher, A. (1991). Bayesian diagnostic probabilities without assuming independence of symptoms. *Methods of Information in Medicine*, 30:15–22.

Geiger, D. (1992). An entropy-based learning algorithm of Bayesian conditional tress. In *Proceedings of the 8th Conference on Uncertainty in Artificial Intelligence*, pages 92–97.

Gillispie, S. and Perlman, M. (2001). Enumerating Markov equivalence classes of acyclic digraph models. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 171–177.

Glover, F. (1998). A template for scatter search and path relinking. In *Artificial Evolution, 3rd European Conference, AE'97*, volume 1363 of *Lecture Notes in Computer Science*, pages 13–54. Springer-Verlag.

Glover, F. and Laguna, M. (1997). *Tabu Search*. Kluwer Academic Publishers.

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

Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caliguri, M., Bloomfield, C., and Lander, E. (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537.

González, C., Lozano, J., and Larrañaga, P. (2000). Analyzing the PBIL algorithms by means of discrete dynamical systems. *Complex Systems*, 12(4):465–479.

Greiner, R. and Zhou, W. (2002). Structural extension to logistic regression: discriminant parameter learning of belief net classifiers. In *Proceedings of the 18th Annual National Conference on Artificial Intelligence*, pages 167–173.

Grossman, D. and Domingos, P. (2004). Learning Bayesian network classifiers by maximizing the conditional likelihood. In *Proceedings of the 21th International Conference on Machine Learning*, pages 361–368.

Guerra-Salcedo, C. and Whitley, D. (1998). Genetic search for feature subset selection. In *Proceedings of the 3rd Annual Genetic Programming Conference*, pages 504–509.

Hamerly, G. and Elkan, C. (2001). Bayesian approaches to failure prediction for disk drives. In *Proceedings of the 18th International Conference on Machine Learning*, pages 202–209.

Hand, D. and You, K. (2001). Idiot's Bayes –not so stupid after all? *International Statistical Review*, 69:385–398.

Hanley, J. and McNeil, B. (1982). The meaning and use of the area under the receiver operating characteristic (ROC) curve. *Radiology*, 143:29–36.

Hastie, T. and Tibshirani, R. (1990). *Generalized Additive Models*. Chapman and Hall.

Heckerman, D., Geiger, D., and Chickering, D. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243.

Heckerman, D., Horvitz, E., and Nathwani, B. (1992). Toward normative expert systems. Part I: the Pathfinder project. *Methods of Information in Medicine*, 31:90–105.

Heckerman, D. and Nathwani, B. (1992). An evaluation of the diagnosis accuracy of Pathfinder. *Computers and Biomedical Research*, 31:90–105.

Henrion, M. (1988). Propagating uncertainty in Bayesian networks by probabilistic logic sampling. In *Proceedings of the 2nd Conference on Uncertainty in Artificial Intelligence*, pages 149–163.

Herskovits, E. and Cooper, G. (1990). Kutató: An entropy-driven system for construction of probabilistic expert systems from database. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pages 54–62.

Holland, J. (1977). *Adaptation in Natural and Artificial Systems*. The University of Michigan Press.

Hosmer, D. and Lemeshow, S. (1989). *Applied Logistic Regression*. John Wiley and Sons.

Howard, R. and Matheson, J. (1981). Influence diagrams. In *Readings on the Principles and Applications of Decision Analysis*, volume 2, pages 721–764. Strategic Decision Group.

Hryceij, T. (1990). Gibbs sampling in Bayesian networks. *Artificial Intelligence*, 46(3):351–363.

Inza, I., Larrañaga, P., Blanco, R., and Cerrolaza, A. (2002a). Filter and wrapper gene selection procedures in dna microarray domains. In *Proceedings of the Workshop 'Bioinformatics and Artificial Intelligence', held within IBERAMIA'02, the VII Iberoamerican Conference on Artificial Intelligence*, pages 23–34.

Inza, I., Larrañaga, P., Blanco, R., and Cerrolaza, A. (2004). Filter versus wrapper gene selection approaches in DNA microarray domains. *Artificial Intelligence in Medicine*, 31(2):91–103.

Inza, I., Larrañaga, P., and Sierra, B. (2001a). Feature subset selection by Bayesian networks: a comparison with genetic and sequential algorithms. *International Journal of Approximate Reasoning*, 27(2):143–164.

Inza, I., Merino, M., Larrañaga, P., Quiroga, J., Sierra, B., and Girala, M. (2001b). Feature subset selection by genetic algorithms and estimation of distributions algorithms. A case study in the survival of cirrhotic patients treated with TIPS. *Artificial Intelligence in Medicine*, 23:187–205.

Inza, I., Sierra, B., Blanco, R., and Larrañaga, P. (2002b). Gene selection by sequential wrapper approaches in microarray cancer class prediction. *Journal of Intelligent and Fuzzy Systems*, 12(1):25–34.

Jensen, A. and Jensen, F. (1996). MIDAS – An influence diagram for management of mildew in winther wheat. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, pages 349–356.

Jensen, C., Kong, A., and Kjærulff, U. (1993). Blocking Gibbs sampling in very large probabilistic expert systems. Technical Report Technical Report R 13-2031, Department of Mathematics and Computer Science, University of Aalborg, Denmark.

Jensen, F. (2001). *Bayesian Networks and Decision Graphs*. Springer-Verlag.

Jensen, F., Kjærulff, U., Olesen, K., and Pedersen, J. (1989). An expert system for control of waste water treatment — a pilot project). Technical report, Judex Datasystemer A/S, Aalborg, Denmark. In Danish.

John, G. (1997). *Enhancements to the Data Mining Process*. Computer Science Department, School of Engineering, Stanford University. PhD thesis.

John, G., Kohavi, R., and Pfleger, K. (1994). Irrelevant features and the subset selection problem. In *Proceedings of the 11th International Conference on Machine Learning*, pages 121–129.

Keller, A. D., Schummer, M., Hood, L., and Ruzzo, W. (2000). Bayesian classification of DNA array expression data. Technical Report UW-CSE-2000-08-01, University of Washington.

Keogh, E. and Pazzani, M. (1999). Learning augmented Bayesian classifiers: a comparison of distributions-based and classification-based approaches. In *Uncertainty 99: The 7th International Workshop on Artificial Intelligence and Statistics*, pages 225–230.

King, R., Feng, C., and Sutherland, A. (1995). STATLOG - comparison of classification algorithms on large real-world problems. *Applied Artificial Intelligence*, 9:289–333.

Kirkpatrick, S., Gerlatt, C., and Vecchi, M. (1983). Optimization by simulated annealing. *Science*, 220:671–680.

Kittler, J. (1978). Feature set search algorithms. *Pattern Recognition and Signal Processing*, pages 41–60.

Kočka, T. and Castelo, R. (2001). Improved learning of Bayesian networks. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 269–276.

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

Kohavi, R. and John, G. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2):273–324.

Kohonen, T. (1995). *Self-Organizing Maps*, volume 30 of *Information Sciences*. Springer-Verlag.

Koller, D. and Sahami, M. (1996). Toward optimal feature selection. In *Proceedings of the 13th International Conference on Machine Learning*, pages 284–292.

Kononenko, I. (1990). Comparison of inductive and naïve Bayesian learning approaches to automatic knowledge acquisition. In *Current Trends in Knowledge Acquisition*.

Kononenko, I. (1991). Semi-naïve Bayesian classifiers. In *Proceedings of the 6th European Working Session on Learning*, pages 206–219.

Kontkanen, P., Myllymäki, P., T.Silander, Tirri, H., and Grünwald, P. (2000). On predictive distributions and Bayesian networks. *Statistics and Computing*, 10:39–54.

Kruskal, W. H. and Wallis, W. A. (1952). Use of ranks in the one-criterion variance analysis. *Journal of American Statistical Association*, 47:583–621.

Kullback, S. and Leibler, R. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, 22:79—86.

Lacave, C. (2002). *Explicación en Redes Bayesianas Causales. Aplicaciones Médicas*. PhD thesis, Universidad Nacional de Educación a Distancia. In Spanish.

Lachenbruch, P. and Mickey, R. (1968). Estimation error rates in discriminant analysis. *Technometrics*, 10:1–11.

Laguna, M. and González-Velarde, J. (1991). A search heuristic for just-in-time scheduling in parallel machines. *Journal of Intelligent Manufacturing*, 2:253–260.

Laguna, M. and Martí, R. (1999). GRASP and path relinking for 2-layer straight line crossing minimization. *INFORMS Journal on Computing*, 11(1):44–52.

Lam, W. and Bacchus, F. (1994). Learning Bayesian belief networks. An approach based on the MDL principle. *Computational Intelligence*, 10(4):269–293.

Langley, P. and Sage, S. (1994). Induction of selective Bayesian classifiers. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 399–406.

Laplace, P. (1814). *Philosophical Essays on Probabilities*. Springer-Verlag. Translation of A. I. Dale from the 5th French edition of 1825.

Larrañaga, P. (2001). An introduction to probabilistic graphical models. In *Estimation of Distribution Algorithms. A New Tool for Evolutionary Optimization*, pages 25–54.

Larrañaga, P. (2003). *Clasificación Supervisada via Modelos Gráficos Probabilísticos*. Research work for the full professor position. In Spanish.

Larrañaga, P., Etxeberria, R., Lozano, J. A., and Peña, J. M. (2000). Combinatorial optimization by learning and simulation of Bayesian networks. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 343–352.

Larrañaga, P., Inza, I., and Cerrolaza, A. (2002). Filter versus wrapper approaches in the selection of accurate genes on DNA microarray domains. In *Proceedings of the 3th Spanish Symposium on Bioinformatics and Computational Biology*, pages 91–92.

Larrañaga, P., Kuijpers, C., Murga, R., and Yurramendi, Y. (1996a). Learning Bayesian network structures by searching for the best ordering with genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 26(4):487–493.

Larrañaga, P. and Lozano, J., editors (2001). *Estimation of Distribution Algorithms. A New Tool for Evolutionary Optimization*. Kluwer Academic Publishers.

Larrañaga, P., Poza, M., Yurramendi, Y., Murga, R., and Kuijpers, C. (1996b). Structure learning of Bayesian networks by genetic algorithms: A performance analysis of control parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):912–926.

Lauritzen, S. (1996). *Graphical Models*. Oxford University Press.

Lauritzen, S. and Spiegelhalter, D. (1988). Local computations with probabilities on graphical structures and their application on expert systems. *Journal of the Royal Statistical Society, Series B*, 50(2):157–224.

Lewis, D. (1992). Feature selection and feature extraction for text categorization. In *Proceedings of Speech and Natural Language Workshop*, pages 212–217.

Lewis, P. (1959). Approximating probability distributions to reduce storage requirements. *Information and Control*, 2:214–225.

Lin, S. and Johnson, K. (2000). *Methods of Microarray Data Analysis*. Kluwer Academic Publishers.

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

Lucas, P. (2004). Restricted Bayesian network structure learning. In *Advances in Bayesian Networks*, pages 217–234. Springer-Verlag.

Mahnig, T. and Mühlenbein, H. (2000). Mathematical analysis of optimization methods using search distributions. In *Proceedings of the 2000 Genetics and Evolutionary Computation Conference*, pages 205–208.

Malinchoc, M., Kamath, P., Gordon, F., Peine, C., Rank, J., and ter Borg, P. (2000). A model to predict survival in patients undergoing transjugular intrahepatic portosystemic shunts. *Hepatology*, 31(4):864–871.

Mani, S., Pazzani, M., and West, J. (1997). Knowledge discovery from a breast cancer database. In *Proceedings of the 6th Conference on Artificial Intelligence in Medicine in Europe*, volume 1211 of *Lecture Notes in Computer Science*, pages 130–133. Springer-Verlag.

Mann, H. and Whitney, D. (1947). On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, 18:50–60.

McCallum, A. and Nigam, K. (1998). A comparison on event models for naïve Bayes text classification. In *Proceedings of the 15th National Conference on Artificial Intelligence, Workshop on Learning for Text Categorization.*

McCarthy, J. (1956). Measures of the value of information. In *Proceedings of the National Academy of Sciences*, pages 645–655.

Meek, C. (2001). Finding a path is harder than finding a tree. *Journal of Artificial Intelligence Research*, 15:383–389.

Meretakis, D. and Wüthrich, B. (1999). Extending naïve Bayes classifiers using long itemsets. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 165–174.

Merino, M. (2004). *Predicción de Mortalidad Precoz tras Implantación de Derivación Portosistémica Percutánea Intrahepática en Pacientes Cirróticos. Aplicación de Métodos de Classificación Supervisada.* PhD thesis, Universidad de Navarra. In Spanish.

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

Minsky, M. (1961). Steps toward artificial intelligence. *Transactions on Institute of Radio Engineers*, 49:8–30.

Mitchell, T. (1997). *Machine Learning.* McGraw-Hill.

Miyahara, K. and Pazzani, M. (2000). Collaborative filtering with the simple Bayesian classifier. In *Proceedings of the Pacific Rim International Conference on Artificial Intelligence*, pages 679–689.

Movellan, J., Wachtler, T., Albright, T., and Sejnowski (2002). Naïve Bayesian coding of color in primary visual cortex. In *Proceedings of the Neural Information Processing Systems 14.*

Mühlenbein, H. (1998). The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5:303–346.

Mühlenbein, H. and Mahnig, T. (1999). FDA: A scalable evolutionary algorithm for optimization of additively decomposed functions. *Evolutionary Computation*, 7(4):353–376.

Mühlenbein, H. and Mahnig, T. (2000). Evolutionary algorithms: from recombination to search distributions. In *Theoretical Aspects of Evolutionary Computing. Natural Computing*, pages 137–176.

Mühlenbein, H. and Paaß, G. (1996). From recombination of genes to the estimation of distributions. In *Parallel Solving from Nature IV*, volume 1411 of *Lecture Notes in Computer Science*, pages 178–187. Springer-Verlag.

Munteanu, P. and Cau, D. (2000). Efficient score-based learning of equivalence classes of Bayesian networks. In *Principles of Data Mining and Knowledge Discovery: 4th European Conference*, number 1910 in Lecture Notes in Computer Science, pages 96–105. Springer-Verlag.

Murphy, A. (1972). Scalar and vector partitions of the probability score (part i), two state situation. *Journal of Applied Meteorology*, 11:273–282.

Myers, J., Laskey, K., and Levitt, T. (1999). Learning Bayesian networks from incomplete data with stochastic search algorithms. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 476–485.

Nadeau, C. and Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, 52:239–281.

Narendra, P. and Fukunaga, K. (1977). A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computer*, C-26(9):917–922.

Neapolitan, R. (1990). *Probabilistic Reasoning in Expert Systems*. John Wiley and Sons.

Neapolitan, R. (2003). *Learning Bayesian Networks*. Prentice Hall.

Ng, A. (1997). Preventing "overfitting" of cross-validation data. In *Proceedings of 14th International Conference on Machine Learning*.

Nordyke, R., Kulikowski, C., and Kulikowski, C. (1971). A comparison of methods for the automated diagnosis of thyroid dysfunction. *Computers and Biomedical Research*, 4:374–389.

Ohmann, C., Moustakis, V., Yang, Q., and Lang, K. (1996). Evaluation of automatic knowledge acquisition techniques in the diagnosis of acute abdominal pain. *Artificial Intelligence in Medicine*, 8:23–36.

Ohmann, C., Yang, Q., Kunneke, M., Stolzing, H., Tohn, K., and Lorenz, W. (1988). Bayes theorem and conditional dependence of symptoms: different models applied to data of upper gastrointestinal bleeding. *Methods of Information in Medicine*, 27:73–83.

Panofsky, H. and Brier, G. (1968). *Some Applications of Statistics to Meteorology*. Pennsylvania State University Press.

Pardo, L. (1997). *Teoría de la Información Estadística*. Hespérides. In Spanish.

Pazzani, M. (1996). Constructive induction of cartesian product attributes. *Information, Statistics and Induction in Science*, pages 66–77.

Pazzani, M. (1997). Searching for dependencies in Bayesian classifiers. In Fisher, D. and Lenz, H., editors, *Artificial Intelligence and Statistics IV, Lecture Notes in Statistics*. Springer-Verlag.

Pazzani, M., Murumatsu, J., and Billsus, D. (1996). Syskill and Webert: identifying interesting web sites. In *Proceedings of the 13th National Conference on Artificial Intelligence*, pages 54–61.

Pearl, J. (1986). Fusion, propagation and structuring in belief networks. *Artificial Intelligence*, 29:241–288.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publishers.

Pérez, O., Marín, F., and Trelles, O. (2002). Weighting and selection of variables on gene expression data by the use of genetic algorithms. Technical Report AC-UMA-03ABR02, Universidad de Málaga.

Pernkopf, F. and O'Leary, P. (2003). Floating search algorithms for structure learning of Bayesian network classifiers. *Pattern Recognition Letters*, 24(15):2839–2848.

Piñana, E., Plana, I., Campos, V., and Martí, R. (2004). GRASP and path relinking for the matrix bandwidth minimization. *European Journal of Operational Research*, 153(1):200–210.

Provost, F., Fawcett, T., and Kohavi, R. (1998). The cases against accuracy estimation for comparing induction algorithms. In *Proceedings of the 15th International Conference on Machine Learning*, pages 445–453.

Pudil, P., Novovičová, J., and Kittler, J. (1994). Floating search methods in feature selection. *Pattern Recognition Letters*, 15(11):1119–1125.

Puerta, J. (2001). *Métodos Locales y Distribuidos para la Construcción de Redes de Creencia Estáticas y Dinámicas*. PhD thesis, Universidad de Granada. In Spanish.

Pugh, R., Murray-Lion, I., Dawson, J., Pictioni, M., and Williams, R. (1973). Transection of the esophagus for bleeding oesophageal varices. *British Journal of Surgery*, 60:646–649.

Quinlan, J. (1986). Induction of decision tress. *Machine Learning*, 1:81–106.

Quinlan, J. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers.

Rao, J. and Shao, A. (1992). Jackknife variance estimation with survey data under hot deck imputation. *Biometrika*, 79:811–822.

Resende, M. and Ribeiro, C. (2003). GRASP and path relinking: recent advances and applications. Technical report, AT&T Labs Research.

Rissanen, J. (1978). Modeling by shortest data description. *Automatics*, 14:465–471.

Robinson, R. (1977). Counting unlabelled acyclic digraphs. In *Lecture Notes in Mathematics: Combinatorial Mathematics V*, pages 28–43. Springer-Verlag.

Romero, T., Larrañaga, P., and Sierra, B. (2004). Learning Bayesian networks in the space of orderings with estimation of distributions algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(4):607–625.

Roure, J. (2002). Incremental learning of tree augmented naïve Bayes classifiers. In *Advances in Artificial Intelligence - 8th Ibero-American Conference on AI*, volume 2527 of *Lecture Notes in Computer Science*, pages 32–41. Springer-Verlag.

Rubinsteins, Y. and Hastie, T. (1997). Discriminative vs informative learning. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, pages 49–53.

Russek, E., Kronmal, R., and Fisher, L. (1983). The effect of assuming independence in applying Bayes' theorem to risk estimation and classification in diagnosis. *Computers and Biomedical Research*, 16:537–552.

Sahami, M. (1996). Learning limited dependence Bayesian classifiers. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 335–338.

Salmerón, A., Cano, A., and Moral, S. (2000). Importance sampling in Bayesian networks using probability trees. *Computational Statistics and Data Analysis*, 34:387–413.

Santana, R. and Ochoa, A. (1999). Dealing with constraints with estimation of distributions algorithms: the univariate case. In *Proceedings of the 2nd Symposium on Artificial Intelligence. Adaptive Systems*, pages 378–384.

Santana, R., Pereira, F., Costa, E., Ochoa, A., Machado, P., Cardoso, A., and Soto, M. (2000). Probabilistic evolutions and the busy beaver problem. In *Late Breaking Papers at the 2000 Genetic and Evolutionary Computation Conference*, pages 261–268.

Saunders, J., Walters, J., Davies, P., and Paton, A. (1981). A 20-year prospective study of cirrhosis. *British Medical Journal*, 282:263–266.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 7(2):461–464.

Shachter, R. and Peot, M. (1990). Simulation approaches to general probabilistic inference on belief networks. In *Proceedings of the 5th Conference on Uncertainty in Artificial Intelligence*, pages 221–234.

Shafer, G. (1996). *Probabilistic Expert Systems*. Society for Industrial and Applied Mathematics.

Shwe, M. and Cooper, G. (1991). An empirical analysis of likelihood-weighting simulation on a large multiply connected medical belief network. *Computers and Biomedical Research*, 24:453–475.

Siedelecky, W. and Sklansky, J. (1988). On automatic feature selection. *International Journal of Pattern Recognition and Artificial Intelligence*, 2:197–220.

Singh, M. and Provan, G. (1996). Efficient learning of selective Bayesian network classifiers. In *Proceedings of the 13th International Conference on Machine Learning*.

Singh, M. and Valorta, M. (1993). An algorithm for the construction of bayesian network structures from data. In *Proceedings of the 9th International Conference on Machine Learning*, pages 259–265.

Somol, P., Pudil, P., and Kittler, J. (2004). Fast branch and bound algorithms for optimal feature selection. *IEEE Transactions on Pattern Analysis and Machine Learning*, 26(7):900–912.

Somol, P., Pudil, P., Novovičová, J., and Paclík, P. (1999). Adaptive floating search methods in feature selection. *Pattern Recognition Letters*, 20(11-13):1157–1163.

Spackman, K. (1989). Signal detection theory: valuable tools for evaluating inductive learning. In *Proceedings of the 6th International Workshop on Machine Learning*, pages 1285–1293.

Spiegelhalter, D. and Knill-Jones, R. (1984). Statistical and knowledge-based approaches to clinical decision-support systems. *Journal of the Royal Statistical Society, Series A*, 147:35–76.

Spirtes, P., Glymour, C., and Scheines, R. (1991). An algorithm for fast recovery of sparse causal graphs. *Social Science Computing Reviews*, 9:62–72.

Spirtes, P., Glymour, C., and Scheines, R. (1993). *Causation, Prediction, and Search*. Lecture Notes in Statistics 81, Springer-Verlag.

Stearns, S. (1976). On selecting features for pattern classifiers. In *Proceedings of the 3th International Conference on Pattern Recognition*, pages 71–75.

Steck, H. (2000). On the use of skeletons when learning in Bayesian networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 558–565.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions (with discussion). *Journal of the Royal Statistical Society B*, 36:111–147.

Tian, J. (2000). A branch and bound algorithm for MDL learning Bayesian networks. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pages 580–588.

Titterington, D., Murray, G., Spiegelhalter, L., Skene, A., Habbema, J., and Gelpke, G. (1981). Comparison of discrimination techniques applied to a complex data set of head injured patients (with discussion). *Journal of the Royal Statistical Society Series A*, 144(2):145–175.

Todd, B. and Stamper, R. (1994). The relative accuracy of a variety of medical diagnostic programs. *Methods of Information in Medicine*, 33:402–416.

Tourassi, G., Frederick, E., Markey, M., and Floyd, C. (2001). Application of the mutual information criterion for feature selection in computer-aided diagnosis. *Medical Physics*, 28(2394–2402).

Tsamardinos, I. and Aliferis, C. (2003). Towards principled features selection: relevancy, filters and wrapper. In *Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics*.

van der Gaag, L. and Renooij, S. (2001). Evaluation scores for probabilistic networks. In *Proceedings of the 13th Belgium-Netherlands Conference on Artificial Intelligence*, pages 109–116.

van der Gaag, L., Renooij, S., Witteman, C., Aleman, B., and Taal, B. (2002). Probabilities for a probabilistic network: a case study in oesophageal cancer. *Artificial Intelligence in Medicine*, 25(2):123–148.

van Dijk, S., van der Gaag, L., and Thierens, D. (2003). A skeleton-based approach to learning Bayesian networks from data. In *Proceedings of the 7th Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 2838 of *Lecture Notes in Computer Science*, pages 132–143. Springer-Verlag.

van Woerkom, A. and Brodman, K. (1961). Statistics for a diagnostic model. *Biometrics*, 17:229–318.

Vinterbo, S. (1999). *Predictive Models in Medicine: Some Methods for Construction and Adaptation*. PhD thesis, Norwegian University of Science and Technology.

Wahba, G. (1988). Comments on 'monotone regression splines in action'. *Statistical Science*, 3:456–458.

Ward, C. (1986). The differential positive rate, a derivative of receiver operating characteristic curves useful in comparing tests and determining decision levels. *Clinical Chemistry*, 32:1428–1429.

Warner, H., Toronto, A., Veasey, L., and Stephenson, R. (1961). A mathematical model for medical diagnosis. Application to congenital heart disease. *Journal of the American Medical Association*, 177:177–184.

Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., and Vapnik, V. (2000). Feature selection for SVMs. In *Proceedings of the Neural Information Processing Systems 12*, pages 668–674.

Wettig, H., Grünwald, P., Roos, T., Myllymäki, P., and Tirri, H. (2002). Supervised learning of Bayesian network parameters made easy. In *Proceedings of the Machine Learning Conference of Belgium and The Netherlands*.

Wolpert, D. (1992). Stacked generalization. *Neural Networks*, 5:241–259.

Wong, M., Lam, W., and Leung, K. (1999). Using evolutionary computation and minimum description length principle for data mining of probabilistic knowledge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(2):174–178.

Xing, E., Jordan, M., and Karp, R. (2001). Feature selection for high-dimensional genomic microarray data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 601–608.

Yang, J. and Honavar, V. (1998). Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 13(2):44–49.

Zaffalon, M. and Hutter, M. (2002). Robust feature selection by mutual information distributions. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pages 577–584.

Zhang, H. and Ling, C. (2001). Learnability of augmented naive Bayes in nominal domains. In *Proceedings of the 18th International Conference on Machine Learning*, pages 617–623.