Available online at www.sciencedirect.com

Computational Statistics & Data Analysis 49 (2005) 998 – 1019

COMPUTATIONAL STATISTICS & DATA ANALYSIS

www.elsevier.com/locate/csda

Confidence intervals for probabilistic network classifiers[☆]

M. Egmont-Petersen[a,*], A. Feelders[a], B. Baesens[b]

[a] *Utrecht University, Institute of Information and Computing Sciences, P. O. Box 80.089, 3508, TB Utrecht, The Netherlands*

[b] *University of Southampton, School of Management, UK*

Received 12 July 2003; received in revised form 28 June 2004
Available online 25 July 2004

Abstract

Probabilistic networks (Bayesian networks) are suited as statistical pattern classifiers when the feature variables are discrete. It is argued that their white-box character makes them transparent, a requirement in various applications such as, e.g., credit scoring. In addition, the exact error rate of a probabilistic network classifier can be computed without a dataset. First, the exact error rate for probabilistic network classifiers is specified. Secondly, the exact sampling distribution for the conditional probability estimates in a probabilistic network classifier is derived. Each conditional probability is distributed according to the bivariate binomial distribution. Subsequently, an approach for computing the sampling distribution and hence confidence intervals for the posterior probability in a probabilistic network classifier is derived. Our approach results in parametric bootstrap confidence intervals. Experiments with general probabilistic network classifiers, the Naive Bayes classifier and tree augmented Naive Bayes classifiers (TANs) show that our approximation performs well. Also simulations performed with the Alarm network show good results for large training sets. The amount of computation required is exponential in the number of feature variables. For medium and large-scale classification problems, our approach is well suited for quick simulations. A running example from the domain of credit scoring illustrates how to actually compute the sampling distribution of the posterior probability.

[☆] Part of this research was presented at the International Workshop on Computational Management Science, Economics, Finance and Engineering, Cyprus, 2003.

[*] Corresponding author. Tel.: +31-30-2534129; fax: +31-30-2513791.
E-mail address: michael@cs.uu.nl (M. Egmont-Petersen).

0167-9473/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.csda.2004.06.018

1. Introduction

Most pattern classifiers are low-level in the sense that they represent the relations between explanatory variables (the features) and the prediction variable (the posterior class distribution. Following the convention in statistical pattern recognition, we use the term posterior probability to indicate the probability $P(C = c_j \mid X = \boldsymbol{x})$) by a compact mathematical function. Typical examples of such classifiers are support vector machines and feed-forward neural networks. Other classifiers—classification trees and the k-nearest neighbor classifier—are transparent in their nature, but the learned representation is often complex (for a discussion see, e.g., Egmont-Petersen and Pelikan, 1999). Probabilistic networks (Bayesian networks or belief networks) are white-box compact statistical models of relations between discrete stochastic variables. In this article, we will show how probabilistic networks can be used as statistical pattern classifiers, how to compute the exact error rate and, most important, derive the sampling distributions for the parameter estimates and for the posterior probability distribution of the classification variable, given the observed feature vector.

Depending on the underlying classification problem, available domain knowledge can take various forms (Egmont-Petersen, 1991). For a problem like credit scoring (Baesens et al., 2002, 2003), bank employees can divide the explanatory variables into subgroups that are related—or monotonous relations may be defined between explanatory variables and the prediction variable (probability of default). In image processing, knowledge of which spatial variations can be expected in a set of images (Cootes et al., 1995) or of the typical spatial interrelations between objects (Archip et al., 2002), may be a priori available. Most pattern classifiers are in fact black boxes in the sense that their parameters cannot be related to domain knowledge. This limitation hampers incorporation of domain knowledge in these pattern classifiers and make them less suited for data mining purposes. Well-known examples of black-box statistical classifiers are neural networks and support vector machines. The weights in feed-forward neural networks (Rumelhart et al., 1986) and the parameters in a support vector machine (Vapnik, 1998) cannot, in general, be related to underlying domain knowledge. More surprisingly, relating the parameters found by discriminant analysis and the thresholds resulting from C4.5 (Quinlan, 1993) to knowledge of the underlying classification problem, is difficult in general. A discrete feature classifier of which the parameters have an intuitive meaning, is logistic regression with discrete explanatory variables (Hosmer, 1984). Its parameters model the likelihood ratios associated with the classes that are to be discriminated. Unfortunately, XOR-like classification problems (Rumelhart et al., 1986) cannot be solved by logistic regression unless an interaction term is included as additional variable.

In this article, we illustrate by an example from credit scoring (Baesens et al., 2002) how probabilistic networks (Lauritzen and Spiegelhalter, 1988; Pearl, 1988) can be tailored for classification problems with *discrete* variables. Ideally, a minimal error-rate pattern classifier computes the posterior probabilities of the class variable *C*, given an observed feature vector $\boldsymbol{x}$. Based on this posterior probability distribution, classification in many applications is based on the winner-takes-all rule which assigns the most likely class label, e.g. c_j, to the case characterized by the feature vector $\boldsymbol{x}$. A probabilistic network represents the multivariate distribution of a set of discrete stochastic variables. Such a network uses a compact graphical representation—we address solely directed acyclic graphs—to specify direct dependence relations between the stochastic variables. The nodes in the graph represent the explanatory variables and the prediction variable. Each arc represents a direct dependency relation between the pair of variables it connects. Each stochastic variable has associated an (un)conditional probability table that specifies the (un)conditional probability distributions corresponding to the different value combinations of its parents (if any). Marginal and conditional independence relations, given the class, are specified by the graph. The independence relations represented by the graph specify how the joint probability distribution is factorised. A probabilistic network lends itself as a white-box statistical classifier, because its parameters constitute either marginal or conditional probabilities. The factorisation of the conditional distribution, $P(C=c \mid X=\boldsymbol{x})$, which is central in statistical pattern classifiers, follows directly from the rules of dependency separation (so-called d-separation, see, e.g., Jensen, 1996).

This article focuses on important aspects of probabilistic network classifiers. We present two novel contributions: (1) the derivation of the exact sampling distribution of the conditional probabilities in a probabilistic network classifier in the case where no prior is being used and (2) an approximation of the sampling distribution of the probabilities, $P(C=c_j \mid X=\boldsymbol{x})$, associated with the (unknown) class membership $c_j,\ \ j=1,\ldots,n_C$, of a case $\boldsymbol{x}$. We derive a parametric bootstrap confidence interval for the conditional probability $P(C=c_j \mid X=\boldsymbol{x})$. The article is structured as follows. After having introduced the mathematical notation, we give an example of a small Bayesian network classifier. It is illustrated how the posterior probability distribution, $P(C=c \mid X=\boldsymbol{x})$, of the class variable, C, is computed by means of the chain rule. We also specify the exact error rate of a probabilistic network classifier. Secondly, the exact sampling distribution for each (un)conditional probability is derived by a frequentist approach. Based on this, an approximate sampling distribution for the posterior probabilities is derived. Thirdly, simulations are conducted with synthetic probabilistic networks and with the well-known Alarm network (Beinlich et al., 1989). The true empiric sampling distribution is compared with the sampling distribution that results from our approach. In the discussion, limitations of our approach and issues for further research are considered.

Our work is related to that of Friedman et al. (Friedman et al., 1999). They use a bootstrapping approach to derive confidence statements about particular features of the network *structure* (e.g., the presence of particular edges or other substructures). In contrast, we focus on establishing confidence bounds on the probabilities computed from a network with a given structure (graph). Hence, our work complements that of Friedman et al.

2. Background

After having introduced the notation, a probabilistic network is briefly defined. Subsequently, the general notion of a minimal error-rate classifier is introduced. An example of how a probabilistic network can be used as a classifier is given.

2.1. Notation

We use capital letters, $A, B, \ldots$, to denote stochastic variables and small letters, $a, b, \ldots$, to indicate particular observations. When required, a subscript is used to indicate a *particular* outcome, e.g., c_j. With n_C, the number of possible outcomes of the variable, C, is indicated. Bold italic letters, $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$, indicate observation vectors, whereas bold capital letters, e.g., $\boldsymbol{X}$, indicate sets of observation vectors. With $P(X = \boldsymbol{x})$ we denote the probability that the set of discrete variables X takes the specific value combination $\boldsymbol{x}$ (In classic statistical pattern recognition (Duda and Hart, 1973), one usually works with continuous stochastic variables and the associated probability density function, $p(X = \boldsymbol{x})$). The joint state space of the variables, $\Omega = \Omega_A \times \Omega_B \cdots$, is finite, which also holds for the number of possible combinations of patterns $\boldsymbol{x} \in \Omega_X$ that can be observed.

2.2. Minimal error-rate classifiers

In statistical pattern recognition, the overall goal is to build classifiers that minimize the total risk R (Duda and Hart, 1973). If we assume that the loss associated with a misclassification is symmetric (the gain of correctly classifying a case equals one minus the loss of classifying the case wrongly), and that the costs of different misclassifications are equal, the error rate ε suffices as assessment criterion (Duda and Hart, 1973). Hence, the goal becomes to learn the pattern classifier that minimizes the error rate, the number of mislabeled cases. Applications in which minimal error-rate classifiers have been employed include prediction of the probability of default on a consumer loan (Baesens et al., 2002), recognition of leukocytes in video images (Egmont-Petersen et al., 2000) and recognition of bone tumors in radiographs (Egmont-Petersen and Pelikan, 1999). Probabilistic classifiers assign class labels $C = c_j$ to cases based on a number of features or measurement values, $X = \boldsymbol{x}$. The conditional probability that the case associated with the vector $\boldsymbol{x}$ belongs to class c_j, is denoted by $P(c_j \mid \boldsymbol{x})$. Hence, application of the winner-takes-all rule to the posterior probability distribution of the class variable C

$$\text{class}(\boldsymbol{x}) = \begin{cases} j: & P(c_j \mid \boldsymbol{x}) > P(c_i \mid \boldsymbol{x}),\ \forall i \neq j, \\ \emptyset: & otherwise, \end{cases} \tag{1}$$

results in the minimal error-rate classifier. Assuming that all cases are assigned a class label (no ties), the fraction of misclassified cases ε is given by (Hashlamoun et al., 1994)

$$\varepsilon = \sum_{\boldsymbol{x} \in \Omega_X} \left(1 - \max_{c \in \Omega_C}(P(c \mid \boldsymbol{x}))\right) P(\boldsymbol{x}). \tag{2}$$

In general, the true posterior probability distribution $P(c \mid \boldsymbol{x})$ is unknown and $P(\boldsymbol{x})$ is represented by a set of labeled cases $\boldsymbol{X}$. The posterior probabilities $P(c \mid \boldsymbol{x})$ need to be estimated from a trained classifier. For a trained classifier that models the multivariate probability distribution $P(\boldsymbol{x})$, the error rate may be computed from

$$\hat{\varepsilon} = \sum_{\boldsymbol{x} \in \Omega_X} \left(1 - \max_{c \in \Omega_C}(\hat{P}(c \mid \boldsymbol{x}))\right) \hat{P}(\boldsymbol{x}). \tag{3}$$

From the definition of a probabilistic network classifier in Section 2.3, it will be clear that its exact error rate can be computed directly from (3), although overfitting may cause $\hat{\varepsilon}$ to be a biased (Assen et al., 2002; Feelders, 2003; Friedman, 1997) estimate of ε. For smaller classifiers that model the complete multivariate probability distribution $P(\boldsymbol{x})$, it is feasible to compute the error rate $\hat{\varepsilon}$ exactly. For example, 10 binary features result in 1024 combinations in Ω_X for which the probability of misclassification needs to be computed. For 20 binary features, about 1 Million combinations of $\boldsymbol{x}$ need to be evaluated.

Some pattern classifiers estimate solely the posterior probabilities $\hat{P}(c \mid \boldsymbol{x})$, but not the multivariate probability distribution $\hat{P}(\boldsymbol{x})$. Such classifiers include feed-forward neural networks and logistic regression. For such classifiers, the error rate has to be estimated from

$$\hat{\varepsilon}(\boldsymbol{X}) = \frac{1}{|\boldsymbol{X}|} \sum_{\boldsymbol{x} \in \boldsymbol{X}} I\left(\text{class}(\boldsymbol{x};\, \hat{P}(c \mid \boldsymbol{x})) \neq l(\boldsymbol{x})\right), \tag{4}$$

with the classifier specific winner-takes-all rule class$(\boldsymbol{x};\ \hat{P}(c \mid \boldsymbol{x}))$, the function $l(\boldsymbol{x})$ specifying the true class label of each vector $\boldsymbol{x} \in \boldsymbol{X}$ and $I(\cdot)$ the indicator function. A representative test set $\boldsymbol{X}$ drawn from $P(\boldsymbol{x})$ is required to compute an unbiased estimate $\hat{\varepsilon}(\boldsymbol{X})$ of the classifier's error rate.

2.3. The probabilistic network classifier

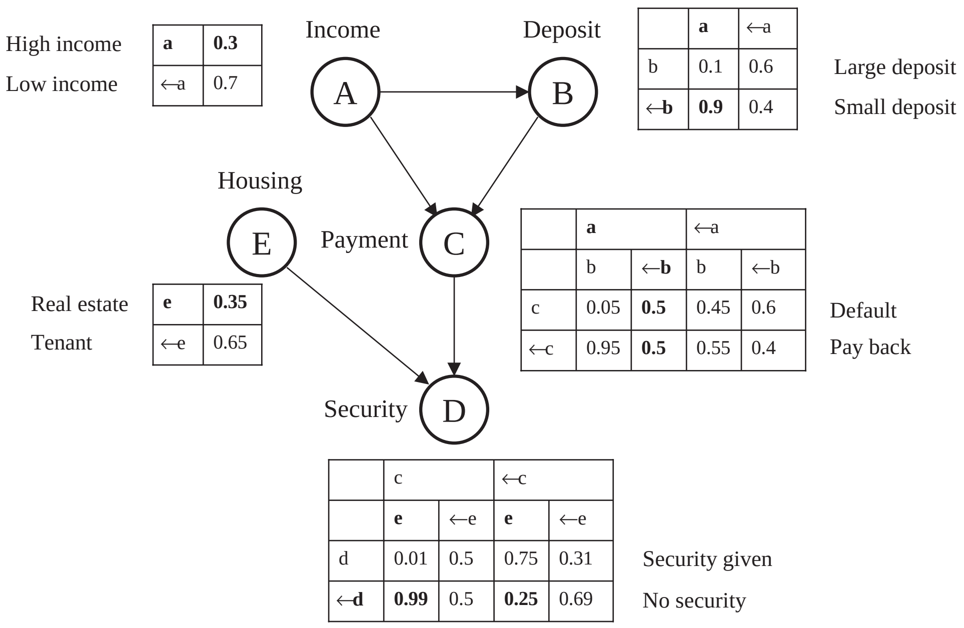
The posterior probability distribution of the classification variable, C, is given by Bayes formula

$$P(c_j \mid \boldsymbol{x}) = \frac{P(\boldsymbol{x} \mid c_j)P(c_j)}{P(\boldsymbol{x})} = \frac{P(\boldsymbol{x} \mid c_j)P(c_j)}{\sum_i P(\boldsymbol{x} \mid c_i)P(c_i)}, \tag{5}$$

with $P(\boldsymbol{x} \mid c_j)$ denoting the class-conditional probability function and $P(c_j)$ the prior probability associated with outcome j of C.

We now define a probabilistic network as a model of a multivariate discrete probability distribution, for an example see Fig. 1. A probabilistic network model M consists of a directed acyclic graph G and a set of probability tables, $\boldsymbol{W}$, $M = (G, \boldsymbol{W})$. The nodes of the directed graph, G, correspond to the set of random variables $D = D_1, \ldots, D_h$ and the edges to direct dependencies between the variables. When $g_{i,j} = 1$, an arc emanates from node i and points to node j. When $g_{i,j} = 0$, there is no arc emanating from node i and pointing to node j. Henceforward, by $\pi(A)$ we denote the direct parents of a node A, $Y = \pi(A)$ (e.g., $\pi(C) = \{A, B\}$ in Fig. 1), and the set of values of the parents by $\boldsymbol{y}_{\pi(A)}$. With $\sigma(A)$, we denote the direct children of node A. The set of probability tables, $\boldsymbol{W} = \{W_1 \ldots W_h\}$, specifies (un)conditional probabilities of the type $P(A = a \mid \pi(A) = \boldsymbol{y}_{\pi(A)})$, $a \in \Omega_A$, $\boldsymbol{y}_{\pi(A)} \in \Omega_{\pi(A)}$.

A probabilistic network classifier represents the class-conditional distributions $P(\boldsymbol{x} \mid c_j)$ implicitly. More specifically, given a complete observation vector $\boldsymbol{x}$, computation of the probability $P(c_j, \boldsymbol{x})$ is efficient because of the independence relations that follow from the graphical structure. Let $\boldsymbol{d} = (c_j, \boldsymbol{x})$ denote the complete observation vector including the true class c_j, corresponding to the stochastic variables $D = (C, X)$. The chain rule



a	**0.3**
←a	0.7

	a	←a
b	0.1	0.6
←b	**0.9**	0.4

e	**0.35**
←e	0.65

	a		←a	
	b	**←b**	b	←b
c	0.05	**0.5**	0.45	0.6
←c	0.95	**0.5**	0.55	0.4

	c		←c	
	e	←e	**e**	←e
d	0.01	0.5	0.75	0.31
←d	**0.99**	0.5	**0.25**	0.69

Fig. 1. Probabilistic network used in the example including the probability tables. The probabilistic network connects the five stochastic variables Income (A), Deposit (B), Payment (C), Security (D) and Housing (E). Based on the values of the four variables specific to an applicant, A, B, D, E, the probability of default can be computed. All variables have to be discrete, see, e.g., Baesens et al. (2002).

(see, e.g., Jensen, 1996)

$$P(D=\boldsymbol{d})=\prod_{k=1}^{h} P(D_k=d_l \mid \pi(D_k)=\boldsymbol{d}_{\pi(D_k)}) \tag{6}$$

makes it feasible to use a probabilistic network as a classifier. From the definition of conditional probability, $P(\boldsymbol{x} \mid c_j)=P(c_j,\boldsymbol{x})/P(c_j)$, and marginalisation over the class variable C, $P(\boldsymbol{x})=\sum_i P(c_i,\boldsymbol{x})$, the posterior probability distribution of C is computed from

$$\begin{aligned} P(c_j \mid \boldsymbol{x}) &= \frac{P(\boldsymbol{x} \mid c_j)P(c_j)}{P(\boldsymbol{x})} = \frac{P(c_j,\boldsymbol{x})/P(c_j)\;P(c_j)}{P(\boldsymbol{x})} \\ &= \frac{P(c_j,\boldsymbol{x})}{\sum_{i=1}^{n_C} P(c_i,\boldsymbol{x})}, \quad j=1,\ldots,n_C. \end{aligned} \tag{7}$$

Little computation is required to calculate the joint probability $P(c_j,\boldsymbol{x})$ for a given probabilistic network as the direct dependencies that follow from the graph G specify the factorisation of $P(c_j,\boldsymbol{x})$. We give a brief example. In the remaining part of the article, we assume that all feature and class variables are binary.

Example 1. In this example, we show how the probabilistic network depicted in Fig. 1 can be used as a statistical classifier. Given the graph G depicted in Fig. 1 connecting the five nodes A, B, C, D and E, the joint probabilities $P(a, \neg b, c, \neg d, e)$ and $P(a, \neg b, \neg c, \neg d, e)$ can be computed by factorization and application of (7). The probability $P(c, \boldsymbol{x})$ factorizes into

$$P(a, \neg b, c, \neg d, e) = P(a)P(\neg b \mid a)P(c \mid a, \neg b)P(\neg d \mid c, e)P(e). \tag{8}$$

The terms can be found by direct table lookup in the probabilistic network (see Fig. 1)

$$P(a, \neg b, c, \neg d, e) = 0.3 \times 0.9 \times 0.5 \times 0.99 \times 0.35 = 0.0467775 \tag{9}$$

and correspondingly

$$P(a, \neg b, \neg c, \neg d, e) = 0.3 \times 0.9 \times 0.5 \times 0.25 \times 0.35 = 0.0118125. \tag{10}$$

Following from (7)

$$P(c \mid a, \neg b, \neg d, e) = \frac{0.0467775}{0.0467775 + 0.0118125} = 0.7984 \tag{11}$$

and $P(\neg c \mid a, \neg b, \neg d, e) = 0.2016$.

When the outcome of one or more variables other than C are unknown (missing), the posterior probability of each unobserved variable, $P(z \mid \boldsymbol{y}), Y \subset X, Z \in X \backslash Y$, can be computed efficiently by highly optimized algorithms, see, e.g., Jensen et al. (1990); Lauritzen and Spiegelhalter (1988). However, in the situation faced in many applications of statistical pattern classifiers, the outcome of each feature variable $\boldsymbol{x}$ is known. Consequently, the posterior probability distribution, $P(c_j \mid \boldsymbol{x})$, can be computed efficiently as illustrated in the example above. Moreover, complex graph-theoretic manipulations including triangulation are avoided when using the chain rule for computing the posterior probability distribution from the probabilistic network classifier.

In the situation addressed here, all feature variables are observed. Hence, the variables that form the *Markov blanket* of C, determine the probability $P(c_j, \boldsymbol{x})$. The Markov blanket is defined relative to a particular node in a probabilistic network. The Markov blanket of C consists of its parents $\pi(C)$, its children $\sigma(C)$ and the parents of the children of C, namely $\pi(\sigma(C))$, other than C. The terms of the chain rule used in (7) can be factorized into a term P_1 solely related to the parents of C, a term, $P_2(C)$, comprising the posterior probability of C given its parents, a term, $P_3(C)$, comprising the children of C, and a term, P_4, comprising the parents of these children that are marginally independent from C. So (7) can at the variable level be rewritten as

$$\begin{aligned} &P(C = c \mid X = \boldsymbol{x}) \\ &\quad = \frac{P_1 P_2(C = c) P_3(C = c) P_4}{P_1 P_2(C = c) P_3(C = c) P_4 + P_1 P_2(C = \neg c) P_3(C = \neg c) P_4}, \end{aligned} \tag{12}$$

$$
\begin{aligned}
P_1 &= \prod_{k_1 \in \pi(C)} P(X_{k_1} \mid \pi(X_{k_1})),\\
P_2(C) &= P(C \mid X_{\pi(C)}),\\
P_3(C) &= \prod_{k_2 \in \sigma(C)} P(X_{k_2} \mid C, \pi(X_{k_2}) \backslash C),\\
P_4 &= \prod_{k_3 \in \pi(\sigma(C)) \backslash \{C \,\cup\, \pi(C) \cup \sigma(c)\}} P(X_{k_3} \mid \pi(X_{k_3}))
\end{aligned}
\tag{13}
$$

for the two-class classifier, with $\pi(\sigma(C))$ denoting the union of the sets of parents with each such set being the parents of a child node of C. Basically, only the terms $P_2(C)$ and $P_3(C)$ vary with C whereas P_1 and P_4 remain constant for each outcome c_j of C. Hence P_1 and P_4 can be divided out and $P(c \mid \boldsymbol{x})$ reduces to

$$
P(c \mid \boldsymbol{x}) = \frac{P_2(c)P_3(c)}{P_2(c)P_3(c) + (1 - P_2(c))P_3(\neg c)}
\tag{14}
$$

for the two-class problem. This result will be used to simplify the computation of the sampling distribution of $\hat{P}(c \mid \boldsymbol{x})$. For later use, we define a topological order t (*topological* refers to the order of the nodes in the graph imposed by the arcs) on the (un)conditional probabilities involved in the computation of $P(c \mid \boldsymbol{x})$ in the probabilistic network classifier as specified in (14)

$$
\begin{aligned}
t \ = \{&P(C = c \mid \boldsymbol{y}_{\pi(C)}),\\
&P(X_1 = x_l \mid c, \boldsymbol{y}_{\pi(X_1)\backslash C}), \ldots, P(X_w = x_q \mid c, \boldsymbol{y}_{\pi(X_w)\backslash C}),\\
&P(X_1 = x_l \mid \neg c, \boldsymbol{y}_{\pi(X_1)\backslash C}), \ldots, P(X_w = x_q \mid \neg c, \boldsymbol{y}_{\pi(X_w)\backslash C})\},
\end{aligned}
\tag{15}
$$

with w the number of children $|\sigma(C)|$ of the classification node. The ordered set t contains $2w + 1$ probability terms for a two-class problem.

Example 1 (Continued). In the probabilististic network classifier in Fig. 1 with the case $\boldsymbol{x} = (a, \neg b, \neg d, e)$, the term $P_1 = P(a)P(\neg b \mid a)$, the second term $P_2(C = c) = P(c \mid a, \neg b)$, $P_2(C = \neg c) = P(\neg c \mid a, \neg b)$ the third term $P_3(C = c) = P(e \mid c, \neg d)$, $P_3(C = \neg c) = P(e \mid \neg c, \neg d)$, and the fourth term $P_4 = P(d)$. For this network, $t = \{P(c \mid a, \neg b), P(\neg d \mid c, e), P(\neg d \mid \neg c, e)\}$, with $w = 1$ because X_1 in (15) corresponds with variable D. So for this network classifier, given the case $\boldsymbol{x}$, (14) becomes

$$
\begin{aligned}
&P(c \mid a, \neg b, \neg d, e)\\
&\quad = \frac{P(c \mid a, \neg b)P(\neg d \mid c, e)}{P(c \mid a, \neg b)P(\neg d \mid c, e) + (1 - P(c \mid a, \neg b))P(\neg d \mid \neg c, e)}.
\end{aligned}
\tag{16}
$$

The posterior probability $P(c \mid a, \neg b, \neg d, e)$ is computed from the three conditional probabilities $P(c \mid a, \neg b)$, $P(\neg d \mid c, e)$ and $P(\neg d \mid \neg c, e)$.

3. Sampling distribution of class variable

Classification is generally performed by a classifier that has been learned from a database $\boldsymbol{D}$. For a probabilistic network classifier, an important distinction should be made between learning the graph G that specifies the direct dependencies between the variables (see e.g., Friedman et al., 1999), and learning the probability tables $\boldsymbol{W}$. In the remaining part of this article, we assume the graph G to be given. The (un)conditional probabilities represented by $\boldsymbol{W}$ are estimated from the complete database $\boldsymbol{D}$ with N cases. Maximum-likelihood estimation from the database, $\max_{\boldsymbol{W}} L(\boldsymbol{W} \mid \boldsymbol{D})$, has a unique solution $\hat{\boldsymbol{W}}$, see further Friedman et al. (1997).

3.1. Sampling distribution of conditional probabilities

The probability distribution of the conditional probability estimate, $\hat{P}(c \mid \boldsymbol{x})$, is derived. In the sequel, no priors, $P(\boldsymbol{W})$, are used resulting in a frequentist probability model. Others have derived approximate confidence intervals for the posterior probabilities, when the Bayesian approach has been taken to sample the parameters of probabilistic networks using Dirichlet priors (Allen et al., 2001). According to our frequentist approach, if a particular combination of variables does not occur in the learning database $\boldsymbol{D}$, the associated (un)conditional probabilities are set to zero. For example, if $(a, \neg b)$ for the network depicted in Fig. 1 does not appear in $\boldsymbol{D}$, $\forall c \in \Omega_C : P(c \mid a, \neg b) = 0$. The consequence of this choice is that feature combinations $\boldsymbol{x}$ that are not contained in the training set, result in the denominator of (7) being zero. Hence, extrapolation is circumvented. In general, the larger the training set for a particular probabilistic network classifier, the smaller is the probability that $P(\boldsymbol{x}) = 0$.

It follows from (7) that the posterior probability $0 \leqslant \hat{P}(c \mid \boldsymbol{x}) \leqslant 1$. The denominator $\hat{P}(\boldsymbol{x}) = \sum_i \hat{P}(c_i, \boldsymbol{x})$ is the estimated probability that the feature vector $\boldsymbol{x}$ is observed, $\hat{P}(\boldsymbol{x}) + \sum_{\boldsymbol{x}' \in (\Omega_X \setminus \boldsymbol{x})} \hat{P}(\boldsymbol{x}') = 1$. Letting N denote the size of the sample $\boldsymbol{D}$, the frequency $N\hat{P}(\boldsymbol{x})$ is binomially distributed, because each case $\boldsymbol{x}$ in the database $\boldsymbol{D}$ that is used to estimate the probability tables $\boldsymbol{W}$, has either the specific combination $\boldsymbol{x}$ or not

$$N(\boldsymbol{x}) \sim B(N, P(\boldsymbol{x})), \tag{17}$$

with $N(\boldsymbol{x})$ indicating the number of occurrences of $\boldsymbol{x}$ in $\boldsymbol{D}$. Similarly, the sampling distribution for the frequency of the estimated numerator of (7), $N\hat{P}(c, \boldsymbol{x})$, is binomially distributed

$$N(c, \boldsymbol{x}) \sim B(N, P(c, \boldsymbol{x})), \tag{18}$$

with $N(c, \boldsymbol{x})$ indicating the number of occurrences of $(c, \boldsymbol{x})$ in $\boldsymbol{D}$. From the derivations in the appendix, it follows that the sampling distribution of the conditional probability $\hat{P}(c \mid \boldsymbol{x})$ is given by the product of two binomial distributions

$$\begin{aligned} &P(\hat{P}(c \mid \boldsymbol{x}), \hat{P}(\boldsymbol{x}) \mid P(c \mid \boldsymbol{x}), P(\boldsymbol{x}), N) \\ &\quad = \binom{k}{m} P(c \mid \boldsymbol{x})^m (1 - P(c \mid \boldsymbol{x}))^{(k-m)} \binom{N}{k} P(\boldsymbol{x})^k (1 - P(\boldsymbol{x}))^{(N-k)}, \\ &\qquad m \leqslant k, \; k \in \{0, \ldots, N\}, \end{aligned} \tag{19}$$

with $\hat{P}(c \mid \boldsymbol{x}) = m/k$, $\hat{P}(\boldsymbol{x}) = k/N$. Eq. (19) is the exact formula for the sampling distribution of any conditional probability in a probabilistic network, and for the bivariate sampling distribution of $\hat{P}(c \mid \boldsymbol{x})$ and $\hat{P}(\boldsymbol{x})$, for given values of $P(c \mid \boldsymbol{x})$, $P(\boldsymbol{x})$ and the sample size N.

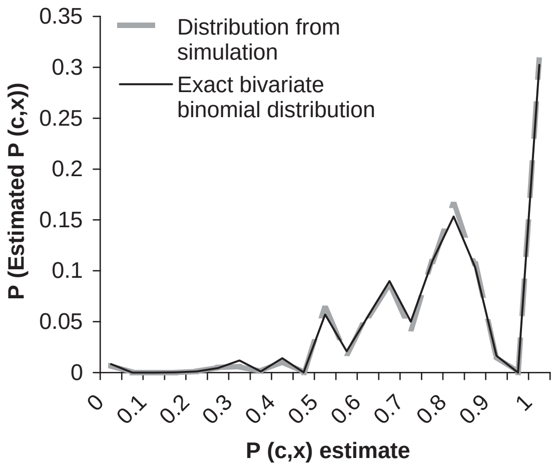


Fig. 2. This graph shows the computed and a simulated empirical sampling distributions of $\hat{P}(c \mid \boldsymbol{x})$, given $P(c \mid \boldsymbol{x}) = 0.7984$ and $P(\boldsymbol{x}) = 0.05859$, $N = 100$ cases in a database, $n = 1000$ sampled databases. The probability $\hat{P}(c \mid \boldsymbol{x})$ and the distribution corresponds to the running example.

Example 1 (Continued). The small example network in Fig. 1 can illustrate the use of (19). The probability

$$P(a, \neg b, \neg d, e) = 0.04678 + 0.01181 = 0.05859 \tag{20}$$

and $P(\neg c \mid a, \neg b, \neg d, e) = 0.7984$. With $p = 0.05859$, $r = 0.7984$, $\boldsymbol{x} = (a, \neg b, \neg d, e)$ and a sample size of $N = 2$, the following probability is obtained

$$\begin{aligned} P(k = 0, m = 0 \mid r, p, \boldsymbol{x}, 2) &= P(\hat{P}(c \mid \boldsymbol{x}) = 0, k = 0) \\ &= \binom{0}{0} r^0 (1-r)^{(0-0)} \binom{2}{0} p^0 (1-p)^{(2-0)} \\ &= 1 \cdot (1 - 0.05859)^2 = 0.8863. \end{aligned} \tag{21}$$

Likewise, the remaining probabilities become

$$\begin{aligned} &P(k = 1, m = 0 \mid r, p, \boldsymbol{x}, 2) = 0.0222, \\ &P(k = 1, m = 1 \mid r, p, \boldsymbol{x}, 2) = 0.0881, \\ &P(k = 2, m = 0 \mid r, p, \boldsymbol{x}, 2) = 1.40 \times 10^{-4}, \\ &P(k = 2, m = 1 \mid r, p, \boldsymbol{x}, 2) = 0.0011, \\ &P(k = 2, m = 2 \mid r, p, \boldsymbol{x}, 2) = 0.0022. \end{aligned} \tag{22}$$

The sum $\sum_k \sum_{m \leqslant k} P(k, m \mid r, p, \boldsymbol{x}, 2) = 1$. Now, $P(\hat{P}(c \mid \boldsymbol{x}) = 0) = \sum_k P(k, m = 0 \mid r, p, \boldsymbol{x}, 2) = 0.90863$, which is the probability that the estimate $\hat{P}(c \mid \boldsymbol{x})$ equals zero given the parameters p, r and a sample size N. The outcome of the variables $\hat{r} = \hat{P}(c \mid \boldsymbol{x})$ and $\hat{p} = \hat{P}(\boldsymbol{x})$ follow from $r = m/k$ and $p = k/N$. Fig. 2 shows this distribution, but for $N = 100$ cases instead of 2.

As the ratio $r = m/k$, it is clear that the number of possible conditional probabilities increases with the sample size N. We will sample the distribution of r into uniformly sized intervals. The distribution of $P_i(\hat{P}(c \mid \boldsymbol{x}) \mid P(c \mid \boldsymbol{x}), P(\boldsymbol{x}), N)$ is sampled into $v + 2$ intervals by

$$P_i\left(\hat{P}(c \mid \boldsymbol{x}) \in \left[\frac{i-0.5}{v+1} \pm \delta\right)\right) = \sum_{k,m \leqslant k \;\mid\; m/k \in [(i-0.5)/(v+1) \pm \delta)} P(k, m \mid r, p, \boldsymbol{x}, N), \tag{23}$$

with $i \in \{1, \ldots, v+2\}$ and $\delta = 1/(2\,(v+1))$. P_i is a multinomial distribution with $v + 2$ outcomes and parameters specified by $P(k, m \mid r, p, \boldsymbol{x}, N)$. In short, the distribution of $\hat{P}(c \mid \boldsymbol{x})$ is denoted by $P_i(\hat{P}(c \mid \boldsymbol{x}) \mid v, \delta)$, for given parameter values r, p, $\boldsymbol{x}$ and N. To sample the distribution of the class variable C, a matrix $\boldsymbol{P}$ is defined in which each row corresponds to a probability term in the ordered set t as defined in (15). Each of the $v + 2$ columns corresponds to an interval probability estimate

$$\boldsymbol{P} = \begin{bmatrix} P_1(\hat{P}(c \mid \boldsymbol{y}_{\pi(C)}) \in [\frac{1}{v+1} \pm \delta)) & \cdots & P_{v+2}(\hat{P}(c \mid \boldsymbol{y}_{\pi(C)}) \in [\frac{v+2-0.5}{v+1} \pm \delta)) \\ P_1(\hat{P}(x_l \mid c, \boldsymbol{y}_{\pi(X_1)\backslash C}) \in [\frac{1}{v+1} \pm \delta)) & \cdots & P_{v+2}(\hat{P}(x_l \mid c, \boldsymbol{y}_{\pi(X_1)\backslash C}) \in [\frac{v+2-0.5}{v+1} \pm \delta)) \\ & \ddots & \\ P_1(\hat{P}(x_q \mid \neg c, \boldsymbol{y}_{\pi(X_w)\backslash\ C}) \in [\frac{1}{v+1} \pm \delta)) & \cdots & P_{v+2}(\hat{P}(x_q \mid \neg c, \boldsymbol{y}_{\pi(X_w)\backslash C}) \in [\frac{v+2-0.5}{v+1} \pm \delta)) \end{bmatrix}. \tag{24}$$

The matrix $\boldsymbol{P}$ contains the sampled distributions of each of the terms in the topological ordering t, row $k \in \{0, \ldots, 2\,w\}$ corresponds to probability term t_k. The parameters of these distributions are denoted by Θ, with the parameters of row k being $\boldsymbol{\theta}_k = \{P(D_k = d \mid \boldsymbol{d}_{\pi(D_k)}), P(\boldsymbol{d}_{\pi(D_k)})\}$. These two parameters, which are denoted r en p in the distribution derived in the appendix, specify the conditional probability of observing d given the values of its parents $\boldsymbol{d}_{\pi(D_k)}$ and the probability of observing this parent combination (Fig. 3).

3.2. *Sampling distribution of class variable*

The sampling distribution in (19) does not capture the distribution of $\hat{P}(c \mid \boldsymbol{x})$ as computed from a Probabilistic network classifier where the classification node has at least one parent and one child. The reason is that each (un)conditional probability distribution in $\boldsymbol{W}$ is normalized, $\hat{P}(D_k = d_l \mid \boldsymbol{d}_{\pi(D_k)}) + \hat{P}(D_k = \neg d_l \mid \boldsymbol{d}_{\pi(D_k)}) = 1$, unless $\hat{P}(\boldsymbol{d}_{\pi(D_k)}) = 0$. In other words, the estimates $\hat{P}(c \mid \boldsymbol{x})$ computed using (7) are solely distributed as specified by (23) when the classification node has only parents. For such a network classifier, the sampling distribution of $\hat{P}(c \mid \boldsymbol{x})$ is given by the single parameter $P(c \mid \boldsymbol{x}_{\pi(C)})$ (The probability $P(\boldsymbol{x}_{\pi(C)})$ and the sample size N still codetermine the sampling distribution of $\hat{P}(c \mid \boldsymbol{x})$), i.e., the special situation where $P_3(C) = 1$ in (14).

When the probability $P(c \mid \boldsymbol{x})$ is composed of more random terms such that $P_3(C) \neq 1$, (23) does not capture the distribution in an appropriate manner. Instead, we will assume that the terms in (15) that form the posterior probability, (14), are distributed independently. More formally, the following conditional independence relations (according to the notation

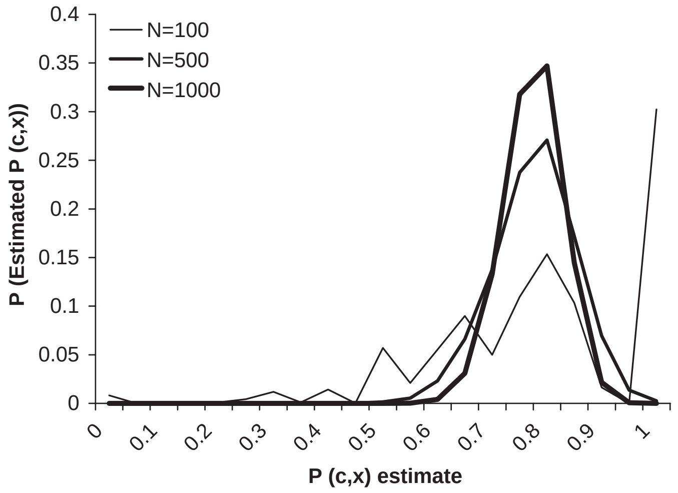


Fig. 3. The graph depicts the true sampling distributions of $\hat{P}(c \mid \boldsymbol{x})$, $N = 100$, 500 and 1000 cases. An increasing sample size N makes estimates close to the true underlying conditional probability 0.7984 more likely.

introduced by Dawid, 1979) are assumed

$$\begin{aligned}
&\hat{P}(D_i = d_l \mid \boldsymbol{y}_{\pi(D_i)}) \quad \perp\!\!\!\perp \quad \hat{P}(D_j = d_m \mid \boldsymbol{z}_{\pi(D_j)}) \mid \boldsymbol{y}_{\pi(D_j)}, \boldsymbol{z}_{\pi(D_i)}, \\
&\qquad\qquad\qquad\qquad\qquad \text{when } i \neq j, \\
&\hat{P}(D_i = d_l \mid \boldsymbol{y}_{\pi(D_i)}) \quad \perp\!\!\!\perp \quad \hat{P}(D_i = d_m \mid \boldsymbol{z}_{\pi(D_i)}) \mid \boldsymbol{y}_{\pi(D_i)}, \boldsymbol{z}_{\pi(D_i)}, \\
&\qquad\qquad\qquad\qquad\qquad \text{when } \boldsymbol{y}_{\pi(D_i)} \neq \boldsymbol{z}_{\pi(D_i)}.
\end{aligned} \tag{25}$$

The simplification implies that all distributions of the parents of each node are sampled independently from each other. In the sequel, we denote the sampling distribution of a probabilistic network classifier by $\hat{P}_B(c \mid \boldsymbol{x})$. The bias introduced by this simplification, which is necessary to make computation of $\hat{P}_B(c \mid \boldsymbol{x})$ tractable, will be investigated experimentally. Consequently, we compute the sampling distribution of the probabilistic network classifier, (7), from

$$\begin{aligned}
&P(\hat{P}_B(c \mid \boldsymbol{x}) \pm \delta \mid v, N) \\
&\quad = \sum_{\boldsymbol{s}} \prod_{k \in \{0,\ldots,2w\}} \boldsymbol{P}_{k,el(s_k)}, \; f(\boldsymbol{s}) \in \{\hat{P}_B(c \mid \boldsymbol{x}) \pm \delta\},
\end{aligned} \tag{26}$$

with the multi-index $\boldsymbol{s} = (s_0 \times \cdots s_k \cdots \times s_{(2w)})^{\mathrm{T}}$ on the variables $k \in \{0, \ldots, 2w\}$, in the topological ordering t defined by (15) and $\delta = 1/(2(v+1))$. The indices $s_k \in \{1/2, 1\ 1/2, \ldots, v + 1/2\}$ represent the estimated posterior probability $\hat{P}_B(c \mid \boldsymbol{x})$, with the function $el(s_k)$ indicating the element in the set represented by s_k. The function f is defined as

$$f(\boldsymbol{s}) = \left(\frac{s_0}{v+1} \cdot \frac{s_1}{v+1} \cdots \frac{s_w}{v+1}\right) \Bigg/ \left(\frac{s_0}{v+1} \cdot \frac{s_1}{v+1} \cdots \frac{s_w}{v+1} + \left(1 - \frac{s_0}{v+1}\right) \frac{s_{(w+1)}}{v+1} \cdots \frac{s_{(2\,w)}}{v+1}\right). \tag{27}$$

The values of $\hat{P}_B(c \mid \boldsymbol{x}) = 1$ are included in the last interval in (26). Finally, define the product term $\boldsymbol{P}(\boldsymbol{s})$ as

$$\boldsymbol{P}(\boldsymbol{s}) = \prod_{k \in \{0,\ldots,2w\}} \boldsymbol{P}_{k,el(s_k)}. \tag{28}$$

We illustrate the use of this formula with the running example.

Example 1 (Continued). The sampling distributions of the three terms in the topological order t, $P(c \mid a, \neg b)$, $P(\neg d \mid c, e)$ and $P(\neg d \mid \neg c, e)$, are obtained from (23). For the first term, $p = P(a, \neg b) = P(a)P(\neg b \mid a) = 0.27$, and $r = 0.5$. Similarly, for the second probability term in t, $p = P(c, e) = P(c)P(e) = 0.172725$ and $r = 0.99$, and for the third term, $p = P(\neg c, e) = P(\neg c)P(e) = 0.177275$ and $r = 0.25$. The resulting distributions for $N = 100$ and $v = 1$ are

$$\boldsymbol{P} = \begin{bmatrix} P(\hat{P}(c \mid a, \neg b) \mid v, N) \\ P(\hat{P}(\neg d \mid c, e) \mid v, N) \\ P(\hat{P}(\neg d \mid \neg c, e) \mid v, N) \end{bmatrix} = \begin{bmatrix} 0.0379 & \mathbf{0.9118} & 0.0503 \\ 0 & 0 & \mathbf{1} \\ \mathbf{0.7655} & 0.2340 & 0.0005 \end{bmatrix}. \tag{29}$$

The probability for each combination of $\boldsymbol{s}$ is being computed by nested for-loops, one loop for each index s_k. For example, for the value $el(\boldsymbol{s}) = (2, 3, 1)^{\mathrm{T}}$, the corresponding probability $\boldsymbol{P}(\boldsymbol{s}) = 0.9118 \times 1 \times 0.7655 = 0.6980$. The complete table becomes

$s_3 = 0.5$	s_2			$s_3 = 1.5$	s_2		$s_3 = 2.5$	s_2	
	0.5	1.5	2.5	0.5	1.5	2.5	0.5	1.5	2.5
$s_1 = 0.5$									
$f(\boldsymbol{s})$	0.1667	0.3750	0.5000	0.0625	0.1667	0.2500	0.0385	0.1071	0.1667
$\boldsymbol{P}(\boldsymbol{s})$	0	0	0.0290	0	0	0.0089	0	0	0
$s_1 = 1.5$									
$f(\boldsymbol{s})$	0.5000	0.7500	**0.8333**	0.2500	0.5000	0.6250	0.1667	0.3750	0.5000
$\boldsymbol{P}(\boldsymbol{s})$	0	0	**0.6980**	0	0	0.2133	0	0	0.0005
$s_1 = 2.5$									
$f(\boldsymbol{s})$	0.8333	0.9375	0.9615	0.6250	0.8333	0.8929	0.5000	0.7500	0.8333
$\boldsymbol{P}(\boldsymbol{s})$	0	0	0.0385	0	0	0.0118	0	0	0

The distribution is ordered into intervals resulting in the final distribution $P(\hat{P}_B(c \mid \boldsymbol{x}) \pm \delta \mid v, N) = (0.0089, 0.2428, 0.7483)$ corresponding to $\hat{P}_B(c \mid \boldsymbol{x}) = 1/(2 \times 3) \pm \delta$, $\hat{P}_B(c \mid \boldsymbol{x}) = 3/(2 \times 3) \pm \delta$ and $\hat{P}_B(c \mid \boldsymbol{x}) = 5/(2 \times 3) \pm \delta$. So the most frequent observed probability $\hat{P}_B(c \mid \boldsymbol{x})$ occurs within the range 0.83 ± 0.17.

Finally, we can now define the parametric bootstrap confidence intervals (Efron and Tibshirani, 1993) of $P_B(c \mid \boldsymbol{x})$. The parameters, $\hat{\boldsymbol{W}}$, in a probabilistic network classifier are estimated from a learning set. These (un)conditional probabilities correspond to an estimate of the parameter matrix, $\hat{\Theta}$, from which the probability distribution $\boldsymbol{P}$ defined in (24) can be computed using the bivariate binomial distribution derived in the appendix. Now either compute or simulate the distribution $\boldsymbol{P}(\boldsymbol{s})$. Define the cumulative distribution of $\boldsymbol{P}(\boldsymbol{s})$ with respect to $f(\boldsymbol{s})$ as $\mathscr{C}_{\boldsymbol{P}}$. The following parametric bootstrap interval results (Efron and Tibshirani, 1993, Chapter 13)

$$[\hat{P}_B(c \mid \boldsymbol{x})_{\mathrm{lo}}, \hat{P}_B(c \mid \boldsymbol{x})_{\mathrm{up}}] = [\mathscr{C}_{\boldsymbol{P}}^{-1}(\alpha), \mathscr{C}_{\boldsymbol{P}}^{-1}(1-\alpha)]. \tag{30}$$

The distribution in each row in $\hat{\boldsymbol{P}}$, defined in (24), is independent from the distribution in each of the other rows. Hence, instead of computing $\boldsymbol{P}(\boldsymbol{s})$ by summing over all possible combinations of the rows in $\boldsymbol{P}(\boldsymbol{s})$, a number of samples can be drawn from each row-distribution. An example is shown on http://www.cs.uu.nl/people/michael/Tutorials/conf-int.html.

4. Experiments

In this section, we report a number of experiments in which the derived sampling distribution of the probability $\hat{P}_B(c \mid \boldsymbol{x})$ is being compared with sampling distributions obtained from simulations. For the first simulation, the network depicted in Fig. 1 is used. Using logic sampling (Henrion, 1988), $n = 1000$ samples, each consisting of $N = 100$ or 500 cases, were obtained from the network. From each sample, the (un)conditional probabilities for a network with the same graph were estimated by maximum likelihood. This network was used to estimate the probability $\hat{P}_B(c \mid \boldsymbol{x})$ corresponding to a particular combination of c and $\boldsymbol{x}$. This resulted in 1000 estimates of $\hat{P}_B(c \mid \boldsymbol{x})$. Our approach presented in the previous section was used to compute the approximation to the distribution of $\hat{P}_B(c \mid \boldsymbol{x})$. The root-mean-squared difference (RMSD) between the sample and the prediction was computed over all the $v + 2$ intervals, as was the absolute difference between the two means (MD). The choice of v is reported in Table 1.

4.1. *Experiments with synthetic networks*

Four different probabilistic network classifiers were used to verify whether the derived sampling distribution of $\hat{P}_B(c \mid \boldsymbol{x})$ approximates the real distributions well. The first network classifier is the one used in the running example throughout this article, see Fig. 1. The specific factorization into independent conditional distributions follows from (16). The second network classifier, Fig. 4, was defined explicitly to disclose the influence of the first assumption specified in (25). The multivariate distribution was designed in a way such that the estimate of the probability $P(c)$ has a high degree of co-variation with the estimate of the probability $P(a)$. Hence, in a sample where less than 95% of the randomly generated cases have the feature value $A = a$, the estimated probability $P(c)$ is also likely to be lower than 0.88. This again influences the conditional probability distributions in the tables of the variables B and D. Recall that this effect is disregarded in our approach in order to make the

Table 1
Experimental results obtained

$n = 1000$ samples	$P(c \mid \boldsymbol{x})$	$P(\boldsymbol{x})$	$N = 100$		$N = 500$	
			RMSD	MD	RMSD	MD
Net1 (40 interv.)						
$\hat{P}(c \mid a, \neg b, \neg d, e)$	0.7984	0.0586	0.0072	0.0003	0.0225	0.0020
$\hat{P}(c \mid \neg a, \neg b, d, e)$	0.0196	0.0300	0.0667	0.0098	0.0328	0.0060
Net2 (40 interv.)						
$\hat{P}(c \mid a, b, d)$	0.9986	0.6550	0.0008	0.0001	0.0000	0.0000
$\hat{P}(c \mid \neg a, \neg b, d)$	0.3684	0.0071	0.0169	0.0032	0.0046	0.0057
Net3 (40 interv.)*						
$\hat{P}(c \mid a, \neg b, d, e)$	0.0182	0.0411	0.0041	0.0002	0.0034	0.0004
$\hat{P}(c \mid \neg a, \neg b, \neg d, \neg e)$	0.5870	0.0690	0.0052	0.0041	0.0082	0.0036
$\hat{P}(c \mid a, \neg b, \neg d, \neg e)$	0.1364	0.0330	0.0062	0.0032	0.0036	0.0015
Net4 (40 interv.)*						
$\hat{P}(c \mid a, \neg b, d, \neg e)$	0.6534	0.0327	0.0067	0.0024	0.0074	0.0019
$\hat{P}(c \mid a, \neg b, d, e)$	0.9754	0.0511	0.0024	0.0007	0.0028	0.0005
$\hat{P}(c \mid \neg a, b, \neg d, e)$	0.1712	0.0946	0.0076	0.0103	0.0072	0.0013

The asterisk * indicates network classifiers where simulations from the matrix $\boldsymbol{P}$ were used to estimate the difference between the estimated and the real sampling distribution.

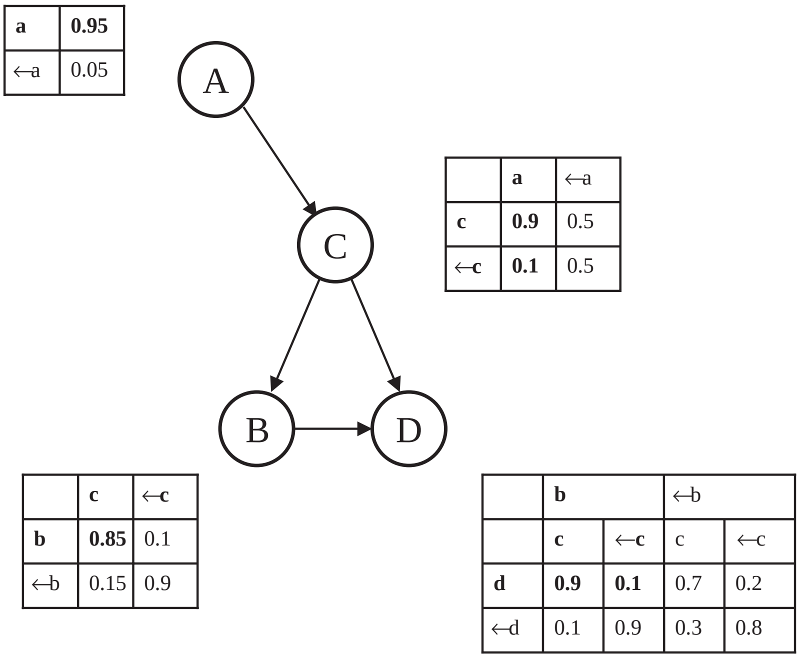
computations tractable. The probability distribution of the second network factorizes into the following components:

$$P(c \mid a, b, d) = \frac{P(c \mid a)P(b \mid c)P(d \mid b, c)}{P(c \mid a)P(b \mid c)P(d \mid b, c) + (1 - P(c \mid a))P(b \mid \neg c)P(d \mid b, \neg c)}. \tag{31}$$

The third network, Fig. 5, is a Naive Bayes classifier, which is probably one of the most frequently used pattern classifiers. Its probability distribution factorizes into the following components

$$P(c \mid a, \neg b, d, e) = \frac{P(c)P(a \mid c)P(\neg b \mid c)P(d \mid c)P(e \mid c)}{P(c)P(a \mid c)P(\neg b \mid c)P(d \mid c)P(e \mid c) + (1 - P(c))P(a \mid \neg c)P(\neg b \mid \neg c)P(d \mid \neg c)P(e \mid \neg c)}. \tag{32}$$

Finally, we constructed a tree-augmented Naive Bayes (TAN) network (Friedman et al., 1997) with the purpose of validating our approach for this type of classifier. The



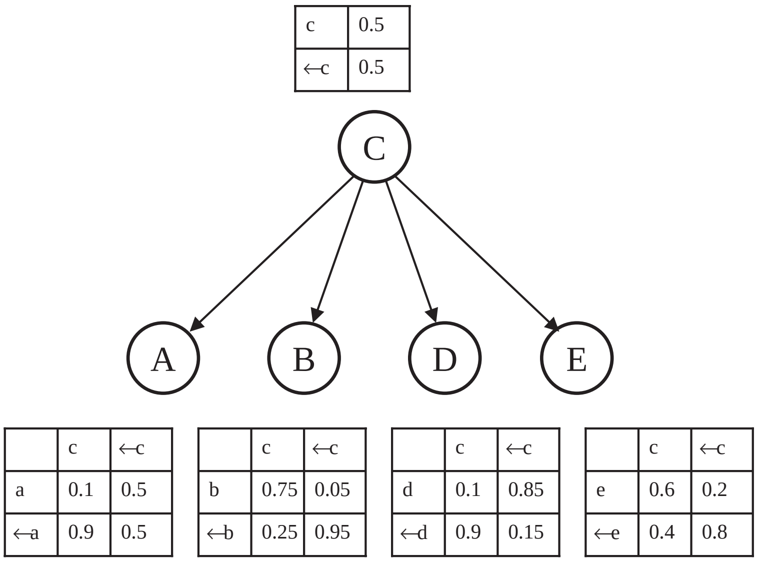
a	**0.95**
¬a	0.05

	a	¬a
c	**0.9**	0.5
¬**c**	**0.1**	0.5

	c	¬**c**
b	**0.85**	0.1
¬b	0.15	0.9

	b		¬b	
	c	¬**c**	c	¬c
d	**0.9**	**0.1**	0.7	0.2
¬d	0.1	0.9	0.3	0.8

Fig. 4. Probabilistic network used in the second experiment.



c	0.5
¬c	0.5

	c	¬c
a	0.1	0.5
¬a	0.9	0.5

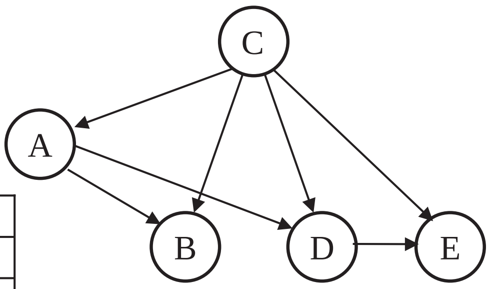
	c	¬c
b	0.75	0.05
¬b	0.25	0.95

	c	¬c
d	0.1	0.85
¬d	0.9	0.15

	c	¬c
e	0.6	0.2
¬e	0.4	0.8

Fig. 5. Probabilistic network, Naive Bayes classifier, used in the third experiment.

c	0.3
¬c	0.7



	c	¬c
a	0.5	0.2
¬a	0.5	0.8

	c		¬c	
	d	¬d	d	¬d
e	0.7	0.6	0.1	0.35
¬e	0.3	0.4	0.9	0.65

	c		¬c	
	a	¬a	a	¬a
b	0.05	0.45	0.7	0.8
¬b	0.95	0.55	0.3	0.2

	c		¬c	
	a	¬a	a	¬a
d	0.5	0.6	0.3	0.5
¬d	0.5	0.4	0.7	0.5

Fig. 6. Probabilistic (TAN) network used in the fourth experiment.

TAN-network is depicted in Fig. 6. The probability $P(c \mid \boldsymbol{x})$ factorizes as follows:

$$
\begin{aligned}
&P(c \mid a, \neg b, d, \neg e)\\
&\quad = \frac{P(c)P(a \mid c)P(\neg b \mid a, c)P(d \mid a, c)P(\neg e \mid c, d)}{\begin{array}{l}P(c)P(a \mid c)P(\neg b \mid a, c)P(d \mid \neg a, c)P(\neg e \mid c, d)+\\ \quad (1-P(c))P(a \mid \neg c)P(\neg b \mid a, \neg c)P(d \mid a, \neg c)P(\neg e \mid \neg c, d).\end{array}}
\end{aligned}
\tag{33}
$$

The results from the experiments shown in Table 1 indicate that the derived sampling distribution of the posterior probability obtained from a probabilistic network classifier, follows the real underlying distribution well. The most imprecise distributions correspond with probabilities close to 0 or 1. The ‘binning’ into intervals causes a discretisation error in the narrow distributions that occur for probabilities close to 0 and 1. The experiments with the network with the skew distribution in Fig. 4, does not seem to give poorer results than the simulations with the three other classifiers.

The approach is exponential in its computational complexity. In practice, when the number of terms in the topological set t exceeds a small number (e.g., (4)), stochastic simulation from each of the distributions in the matrix $\boldsymbol{P}$, is necessary to estimate the sampling distribution of $P_B(c \mid \boldsymbol{x})$. As each row in $\boldsymbol{P}$ represents an independent distribution, such a sampling from each row distribution is straightforward.

Table 2
Experimental results obtained with the Alarm network

	$P(a \mid \boldsymbol{x})$	$P(\boldsymbol{x})$	$N = 100$		$N = 500$		$N = 1000$	
			RMSD	MD	RMSD	MD	RMSD	MD
$\hat{P}(a \mid b, c, d_1, e_1)$	0.3356	0.0056	0.0890	0.0633	0.0177	0.0225	0.0165	0.0082
$\hat{P}(a \mid b, \neg c, d_2, e_1)$	0.9954	0.0085	0.0298	0.1494	0.0076	0.0028	0.0223	0.0030
$\hat{P}(a \mid b, c, d_2, e_2)$	1.0000	0.6094	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
$\hat{P}(a \mid b, \neg c, d_1, e_1)$	0.5025	0.0019	0.1409	0.6941	0.0631	0.2376	0.0274	0.1204
$\hat{P}(a \mid \neg b, \neg c, d_1, e_1)$	0.0011	0.0084	0.0917	0.1789	0.0015	0.0036	0.0000	0.0000

All sampling distributions of $\hat{P}(a \mid \boldsymbol{x})$ where simulations from the matrix $\boldsymbol{P}$.

4.2. *Experiment with the ALARM network*

Experiments with the well-known Alarm network (Beinlich et al., 1989) were performed to give an indication of how the sampling distribution is approximated for a large (realistic) probabilistic network classifier. Essentially, the Alarm network is a classifier because the outcome of one particular node, corresponding to the variable *Lvfailure*, is being monitored. The markov blanket of *Lvfailure* contains the four feature variables *History*, *Hypovolemia*, *Lvedvolume* and *Strokevolume*. We rename the variables in the following way: $A = Lvfailure$, $B = History$, $C = Hypovolemia$, $D = Lvedvolume$ and $E = Strokevolume$. We performed simulation experiments with five different combinations of the four feature variables, see Table 2. The variables A, B and C are binary whereas the feature variables D and E have three outcomes.

The Alarm network is difficult to sample because the distribution is skew. In more than 60% of the cases, the combination (b, c, d_2, e_2) occurs resulting in a probability $\hat{P}(a \mid b, c, d_2, e_2) \approx 1$. In many samples, one or more conditional probabilities tend to be either 0 or 1. As the sample size grows, the estimated sampling distribution converges towards the true underlying distribution. The largest discrepancy was observed for the rare combination $b, \neg c, d_1, e_1$. As would be expected, the sampling distributions with the smallest bias are obtained for learning sets with a size $N = 1000$.

5. Discussion

In this article, we described how a probabilistic network can be used as a minimal error-rate statistical classifier. The exact error rate was given and related to error rate estimates common in statistical pattern recognition. Secondly, we derived the exact sampling distribution of the conditional probabilities in a probabilistic network classifier. Based on this result, an approximate sampling distribution is derived for the conditional probability $P(C = c \mid X = \boldsymbol{x})$ with c the class label and $\boldsymbol{x}$ a complete feature vector.

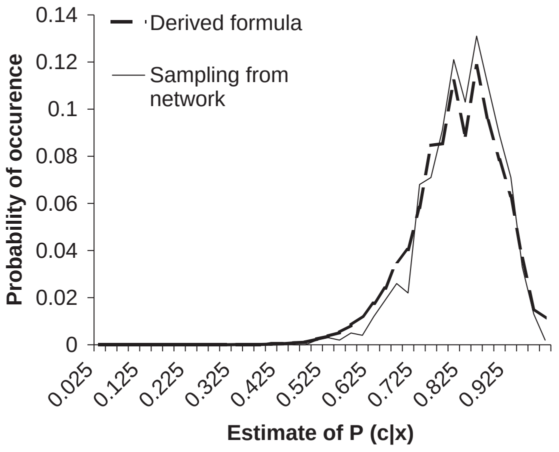


Fig. 7. Prediction by the theoretical formula for the network used in the running example, true posterior probability $P(c \mid \boldsymbol{x}) = 0.7984$.

The experiments we performed indicate that the sampling distribution computed by our approach corresponds well with the simulated one. Solely when the true probability $P(c \mid \boldsymbol{x})$ lies outside the interval $(0.05, 0.95)$, our approach results in slightly skew estimated sampling distributions. The key to this positive result seems to be that we use the exact bivariate binomial distributions of the underlying conditional probabilities to estimate the sampling distribution of $\hat{P}(c \mid \boldsymbol{x})$. It is our advice not to use the Gaussian approximation to the binomial distribution, unless the sample size becomes large. More specifically, the product $P(\boldsymbol{x}) \cdot N$ determines the fraction of cases in a database $\boldsymbol{D}$ that actually contribute to estimating the conditional probability $P(c \mid \boldsymbol{x})$. Our simulation experiments with the Alarm network suggest that the sampling distribution (and hence confidence intervals) become reliable when $P(\boldsymbol{x}) \cdot N > 5$. The more variables and arcs a graph includes, the more conditional probabilities need to be estimated. It is also clear from our experiments (see also Fig. 7), that the approximation usually made that the sampling distribution is symmetric (see, e.g., the Bayesian approach by Allen et al., 2001), does not hold, even when $P(c \mid \boldsymbol{x})$ is rather close to $1/2$.

Probabilistic networks, in general, suffer from the *curse of dimensionality*. A large number of variables or a graph where one or more nodes have many parents, implies an increasing variance of the parameter estimates. This follows directly from the bivariate binomial distribution derived in the appendix. When an increasing number of parameters come into play, while the size N of the learning database remains constant, the variance of the probability estimate $\hat{P}(c \mid \boldsymbol{x})$ inevitably grows. Our approach to computing the sampling distribution can be used to control the amount of variance in the sense that the effect of changing the size of the training set N can be estimated. Others have used a Bayesian approach based on Dirichlet priors to smooth the conditional probability estimates. Dirichlet priors may be seen as an aid for controlling the curse of dimensionality through regularization.

We have restricted our approach to binary classification problems. The approach generalizes directly to feature variables with more than two outcomes, as was illustrated by the experiments with the Alarm network. Our approach can be extended to problems with more than two classes, but the computational complexity will grow exponentially. For the two-class problem, the posterior probability distribution can be simulated using our approach when the number of independent components in the ordered set t exceeds say 4. As shown by our experiments with the Naive bayes and the TAN classifiers as well as with the Alarm network, such a simulation is unproblematic. Each sampled distribution (row) in the matrix $\boldsymbol{P}$ defined in (24) is assumed to be independent from the others. Applying the standard uniform distribution `rnd`—available in most modern numerical software—to each cumulative distribution, e.g., $P_i(\hat{P}(x_l \mid c, \boldsymbol{y}_{\pi(X_{k-1})\setminus C}) \in [0 \pm \delta))$, results in an index s_k and a probability $\boldsymbol{P}_{k,el(s_k)}$.

Finally, we would like to mention the smoothing effect that occurs when the feature variables are children of the class variable, rather than parents. If say four boolean variables were used as parents, resulting in a large table W_C with 2^4 entries, the exact formula for $P(\hat{r}, \hat{p} \mid r, p, \boldsymbol{x}, N)$ derived in the appendix indicates the sampling distribution of the class variable C. The result is a rather ragged distribution (similar to the one depicted in Fig. 3), unless either the number of cases or the probability of observing this particular combination of the parents becomes really large. As soon as (un)conditional probabilities are sampled independently, as is the case in all the networks used in our experiments, a more smooth sampling distribution of $\hat{P}(c \mid \boldsymbol{x})$ results.

6. Conclusion

In this article, we showed how a general probabilistic network can be used as a statistical classifier, resulting in a probabilistic network classifier. We specified how the exact error rate can be computed for such a probabilistic network classifier, without the need of a training set. We then derived the exact sampling distribution for the conditional probability estimates in a probabilistic network. Subsequently, an approach for computing the sampling distribution and hence confidence intervals for the conditional probability $\hat{P}(c \mid \boldsymbol{x})$ in a probabilistic network classifier was derived. When the confidence intervals are accurate enough, it is also possible to test different hypotheses regarding the probability of class membership of a case $\boldsymbol{x}$. Experiments revealed that our approximation performs well on general probabilistic network classifiers (where the class node has parents as well as children), on the Naive Bayes classifier and on tree augmented Naive Bayes networks. We also tested our approach on the well-known Alarm network. The amount of computation required is exponential in the number of feature variables. For medium and large scale classification problems, our approach is well-suited for quick simulations.

In the future, several issues can be investigated further. It would be desirable to establish a bound on the approximation error that is made by our approach. Such a bound could hopefully indicate types of underlying probability distributions for which the approximation derived here performs well. It would also be interesting to extend our approach to incorporate priors on the parameters.

Appendix A

Define $p = P(\boldsymbol{x})$, $q = P(c, \boldsymbol{x})$ and $r = P(c \mid \boldsymbol{x})$. The sampling distributions of $\hat{p}$ and $\hat{q}$ become $P(\hat{p}\, N \mid p, N) = B(N, p)$ and $P(\hat{q}\, N \mid q, N) = B(N, q)$. Solely for sufficiently large probabilities and sample sizes, $\hat{p}N > 50$ (Egmont-Petersen et al., 1994), the continuous Gaussian distribution is suited as approximation. Consequently, we derive the distribution of $\hat{r}$ from the exact binomial distributions $B(N, p)$ and $B(N, q)$. To simplify the notation, we write $P(\hat{p} \mid p, N)$ meaning $P(\hat{p}N \mid p, N)$. The distribution of $\hat{p}$ is given by

$$P(\hat{p} \mid p, N) = \binom{N}{k} p^k (1-p)^{(N-k)}, \quad k \in \{0, \ldots, N\}, \tag{A.1}$$

with $\hat{p} = k/N$ and $\binom{N}{k}$ the binomial coefficient. Similarly, the distribution of $\hat{q}$ becomes

$$P(\hat{q} \mid q, N) = \binom{N}{m} q^m \ (1-q)^{(N-m)}, \quad m \in \{0, \ldots, N\}, \tag{A.2}$$

with $\hat{q} = m/N$. It is clear that p and q are statistically dependent, $p = 0 \Rightarrow q = 0$. As no priors are used, $\hat{p}$ and $\hat{q}$ may be zero. Moreover, when $\hat{p}N$ becomes small, the resulting courser sampling scheme limits the possible estimates of $\hat{q}$.

Our goal is to compute $P(\hat{r} \mid r, p, \boldsymbol{x}, N)$, $\hat{r} \in (0, 1)$, from $\boldsymbol{D}$ with $r = q/p$. The conditional sampling distribution $P(\hat{r} \mid r, \hat{p}, \boldsymbol{x}, N)$ is given by

$$P(\hat{r} \mid r, \hat{p}, \boldsymbol{x}, N) = \binom{k}{m} r^m \ (1-r)^{(k-m)}, \quad m \leqslant k, \tag{A.3}$$

with $r = P(c \mid \boldsymbol{x})$, $\hat{r} = m/k$ and $\hat{p} = k/N$. The variable k denotes the actual number of cases in a sample database $\boldsymbol{D}$ with the combination $\boldsymbol{x}$, whereas m denotes the actual number of cases in $\boldsymbol{D}$, with the combination $\boldsymbol{x}$, that belong to the class c. In general, the chain rule allows us to write the bivariate distribution $P(\hat{r}, \hat{p} \mid r, p)$ as $P(\hat{r} \mid r, \hat{p}, p)P(\hat{p} \mid r, p)$, and because $\hat{r}$ is *independent* from p given $\hat{p}$ (the posterior probability $\hat{r}$ is estimated from the subsample of cases in $\boldsymbol{D}$ with the feature vector $\boldsymbol{x}$, i.e., $\hat{p}$ determines the distribution of $\hat{r}$), $P(\hat{r}, \hat{p} \mid r, p) = P(\hat{r} \mid r, \hat{p})P(\hat{p} \mid r, p)$. Finally, as $\hat{p}$ is independent from r (the probability of observing cases with the feature vector $\boldsymbol{x}$ marginalises over the class variable, hence the relation between $P(c_j \mid \boldsymbol{x})$ and $P(c_i \mid \boldsymbol{x})$ does not influence $p = \sum_i P(c_i, \boldsymbol{x}) = P(\boldsymbol{x})$), it follows that $P(\hat{r}, \hat{p} \mid r, p) = P(\hat{r} \mid r, \hat{p})P(\hat{p} \mid p)$. Consequently, the bivariate distribution $P(\hat{r}, \hat{p} \mid r, p, \boldsymbol{x}, N)$ becomes

$$P(\hat{r}, \hat{p} \mid r, p, \boldsymbol{x}, N) = P(\hat{r} \mid r, \hat{p}, \boldsymbol{x}, N)P(\hat{p} \mid p, \boldsymbol{x}, N). \tag{A.4}$$

The expression on the right-hand side of (A.4) is a product of two binomial distributions

$$\begin{aligned} P(\hat{r}, \hat{p} \mid r, p, \boldsymbol{x}, N) \quad = \quad & \binom{k}{m} r^m (1-r)^{(k-m)} \binom{N}{m} p^k \ (1-p)^{(N-k)}, \\ & m \leqslant k, \ \ k \in \{0, \ldots, N\}, \end{aligned} \tag{A.5}$$

with $\hat{r} = m/k$, $\hat{p} = k/N$. Equation (A.5) is the exact formula for the bivariate sampling distribution of $\hat{P}(c \mid \boldsymbol{x})$ and $\hat{P}(\boldsymbol{x})$, for given values of $P(c \mid \boldsymbol{x})$, $P(\boldsymbol{x})$ and the sample size N.

References

Allen, T.V., Greiner, R., Hooper, P., 2001. Bayesian error-bars for belief net inference. In: Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence (UAI-01). Seattle, pp. 522–529.

Archip, N., Erard, P., Egmont-Petersen, M., Haefliger, J., Germond, J., 2002. A knowledge-based approach to automatic detection of the spinal cord in ct images. IEEE Trans. Med. Imaging 21 (12), 1504–1516.

Assen, H.v., Egmont-Petersen, M., Reiber, J., 2002. Accurate object localization in gray level images using the center of gravity measure; accuracy versus precision. IEEE Trans. Image Process. 11 (12), 1379–1384.

Baesens, B., Egmont-Petersen, M., Castelo, R., Vanthienen, J., 2002. Learning bayesian network classifiers for credit scoring using markov chain monte carlo search. In: Proceedings of the International Conference on Pattern Recognition. IEEE Computer Society, Piscataway, pp. 49–52.

Baesens, B., Setiono, R., Mues, C., Vanthienen, J., 2003. Using neural network rule extraction and decision tables for credit-risk evaluation. Manage. Sci. 49 (3), 312–329.

Beinlich, I., Suermondt, G., Chavez, R., Cooper, G., 1989. The alarm monitoring system: a case study with two probabilistic inference techniques for belief networks. In: Proceedings Second European Conference on AI and Medicine. Springer, Berlin, pp. 247–256.

Cootes, T., Taylor, C., Cooper, D., Graham, J., 1995. Active shape models—their training and application. Comput. Vision Image Understanding 61 (1), 38–59.

Dawid, A., 1979. Conditional independence in statistical theory. J. Roy. Statist. Soc. Ser. B 41 (1), 1–31.

Duda, R., Hart, P., 1973. Pattern Classification and Scene Analysis, Wiley, New York.

Efron, B., Tibshirani, R., 1993. An Introduction to the Bootstrap, Chapman & Hall, New York.

Egmont-Petersen, M., 1991. Mental models as cognitive entities. In: Mayoh, B. (Ed.), Proceedings of the Scandinavian Conference on Artificial Intelligence. IOS Press, pp. 205–210.

Egmont-Petersen, M., Pelikan, E., 1999. Detection of bone tumours in radiographs using neural networks. Pattern Anal. Appl. 2 (2), 172–183.

Egmont-Petersen, M., Talmon, J., Brender, J., NcNair, P., 1994. On the quality of neural net classifiers. Artif. Intell. Med. 6 (5), 359–381.

Egmont-Petersen, M., Schreiner, U., Tromp, S., Lehmann, T., Slaaf, D., Arts, T., 2000. Detection of leukocytes in contact with the vessel wall from in vivo microscope recordings using a neural network. IEEE Trans. Biomed. Eng. 47 (7), 941–951.

Feelders, A., 2003. Statistical concepts. In: Berthold, M., Hand, D. (Eds.), Intelligent Data Analysis, 2nd Edition. Springer, Berlin, pp. 17–68.

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

Friedman, N., Geiger, D., Goldszmidt, M., 1997. Bayesian network classifiers. Mach. Learn. 29 (2–3), 131–163.

Friedman, N., Goldszmidt, M., Wyner, A., 1999. Data analysis with bayesian networks: a bootstrap approach. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence. pp. 206–215.

Hashlamoun, W.A., Varshney, P.K., Samarasooriya, V.N.S., 1994. A tight upper bound on the bayesian probability of error. IEEE Trans. Pattern Anal. Mach. Intell. 16 (2), 220–224.

Henrion, M., 1988. Propagating uncertainty in bayesian networks by probabilistic logic sampling. In: Lemmer, J., Kanal, L. (Eds.), Uncertainty in Artificial Intelligence 2, Vol. 7. North-Holland, Amsterdam, pp. 149–163.

Hosmer, D., 1984. Applied Logistic Regression, Wiley, New York.

Jensen, F., Lauritzen, S., Olesen, K., 1990. Bayesian updating in causal probabilistic networks by local computations. Comput. Statist. Q. 4, 269–282.

Jensen, F.V., 1996. An Introduction to Bayesian Networks, UCL Press and Springer, London.

Lauritzen, S., Spiegelhalter, D., 1988. Local computations with probabilities on graphical structures and their application to expert systems. J. Roy. Statist. Soc. Ser. B 50 (2), 157–224.

Pearl, J., 1988. Probabilistic Reasoning in Intelligent Systems, Morgan-Kaufmann, San Mateo.

Quinlan, J., 1993. C4.5: Programs for Machine Learning, Morgan-Kaufmann, San Mateo.

Rumelhart, D., Hinton, G., Williams, R., 1986. Learning Internal Representations by Error Propagation. MIT Press, Cambridge, pp. 319–362.

Vapnik, V., 1998. Statistical Learning Theory, Wiley, New York.