International Journal of Approximate Reasoning 68 (2016) 153–163

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

Learning extended tree augmented naive structures ☆

Cassio P. de Campos [a,*], Giorgio Corani [b], Mauro Scanagatta [b], Marco Cuccu [c], Marco Zaffalon [b]

[a] Queen's University Belfast, UK
[b] Istituto Dalle Molle di Studi sull'Intelligenza Artificiale, Switzerland
[c] University of Lugano, Switzerland

ARTICLE INFO

Article history:
Received 6 January 2015
Received in revised form 16 April 2015
Accepted 21 April 2015
Available online 23 April 2015

Keywords:
Bayesian networks
Structure learning
Classification
Tree augmented naive Bayes
Edmonds' algorithm

ABSTRACT

This work proposes an extended version of the well-known tree-augmented naive Bayes (TAN) classifier where the structure learning step is performed without requiring features to be connected to the class. Based on a modification of Edmonds' algorithm, our structure learning procedure explores a superset of the structures that are considered by TAN, yet achieves global optimality of the learning score function in a very efficient way (quadratic in the number of features, the same complexity as learning TANs). We enhance our procedure with a new score function that only takes into account arcs that are relevant to predict the class, as well as an optimization over the equivalent sample size during learning. These ideas may be useful for structure learning of Bayesian networks in general. A range of experiments shows that we obtain models with better prediction accuracy than naive Bayes and TAN, and comparable to the accuracy of the state-of-the-art classifier averaged one-dependence estimator (AODE). We release our implementation of ETAN so that it can be easily installed and run within Weka.

1. Introduction

Classification is the problem of predicting the *class* of a given object on the basis of some attributes (*features*) of it. A classical example is the iris problem by Fisher: the goal is to correctly predict the *class*, that is, the species of iris on the basis of four features (sepal and petal length and width). In the Bayesian framework, classification is accomplished by updating a prior density (representing the beliefs before analyzing the data) with the likelihood (modeling the evidence coming from the data), in order to compute a posterior density, which is then used to select the most probable class.

The naive Bayes classifier [2] is based on the assumption of stochastic independence of the features given the class; since the real data generation mechanism usually does not satisfy such a condition, this introduces a bias in the estimated probabilities. Yet, at least under the zero-one accuracy, the naive Bayes classifier performs surprisingly well [2,3]. Reasons for this phenomenon have been provided, among others, by Friedman [4], who proposed an approach to decompose the misclassification error into bias error and variance error; the bias error represents how closely the classifier approximates the target function, while the variance error reflects the sensitivity of the parameters of the classifier to the training sample.

☆ This paper is an extended and revised version of material originally presented in [1].

* Corresponding author.
E-mail addresses: c.decampos@qub.ac.uk (C.P. de Campos), giorgio@idsia.ch (G. Corani), mauro@idsia.ch (M. Scanagatta), marco.cuccu@usi.ch (M. Cuccu), zaffalon@idsia.ch (M. Zaffalon).

http://dx.doi.org/10.1016/j.ijar.2015.04.006
0888-613X/

Low bias and low variance are two conflicting objectives; for instance, the naive Bayes classifier has high bias (because of the unrealistic independence assumption) but low variance, since it requires to estimate only a few parameters. A way to reduce the naive Bayes bias is to relax the independence assumption using a more complex graph, like a tree-augmented naive Bayes (TAN) [5]. In particular, TAN can be seen as a Bayesian network where each feature has the class as parent, and possibly also a feature as second parent. In fact, TAN is a compromise between general Bayesian networks, whose structure is learned without constraints, and the naive Bayes, whose structure is determined in advance to be naive (that is, each feature has the class as the only parent). TAN has been shown to outperform the naive Bayes classifier in a range of experiments [5–7].

In this paper we develop an extension of TAN that allows it to have (i) features without the class as parent, (ii) multiple features with only the class as parent (that is, building a forest), (iii) features completely disconnected (that is, automatic feature selection). While the most common usage of this model is traditional classification, it represents a novel way to learn Bayesian network structures that extend current polynomial-time state-of-the-art methods. In this respect, learning TANs can be seen as the best low-complexity algorithm for exact learning of Bayesian networks. In spite of that, our extension of TAN can also be used as a component of a graphical model suitable for multi-label classification [8].

Extended TAN (or simply ETAN) is learned in quadratic time in the number of features, which is essentially the same computational complexity as that of TAN (our actual ETAN implementation has a $\log^*$ term, which can be neglected for any reasonable number of features). The goodness of each (E)TAN structure is assessed through a decomposable and likelihood equivalent score, such as the Bayesian Dirichlet likelihood equivalent uniform (BDeu) [9–12]. Because ETAN's search space of structures includes that of TAN, the score of the best ETAN is always equal or superior to that of the best TAN. ETAN thus provides a better fit. However, it is well known that this fit does not necessarily imply higher classification accuracy [13]. To improve on ETAN as a classifier, we propose a new score function that takes into account only features that are not (conditionally) independent of the class (given the other features). ETAN under this new scoring idea is empirically shown to produce higher accuracy than Naive Bayes, TAN, and itself (using the standard BDeu).

We perform extensive experiments with these classifiers. We empirically show that ETAN yields in general better zero-one accuracy and log loss than TAN and naive Bayes (where log loss is computed from the posterior distribution of the class given features). Log loss is relevant in cases of cost-sensitive classification [14,15]. We also study the possibility of optimizing the equivalent sample size of ETAN [16], which makes it perform similar to the averaged one-dependence estimator (AODE).

This paper is divided as follows. Section 2 introduces notation and defines the problem of learning Bayesian networks and the classification problem. Section 3 presents our new classifier and an efficient algorithm to learn it from data. Section 4 describes our experimental setting and discusses on empirical results. Finally, Section 5 concludes the paper and suggests possible future work.

2. Learning TANs and classification

The classifiers that we discuss in this paper are all subcases of a Bayesian network. A Bayesian network represents a joint probability distribution over a collection of categorical random variables. It can be defined as a triple $(\mathcal{G}, \mathcal{X}, \mathcal{P})$, where $\mathcal{G} = (V_\mathcal{G}, E_\mathcal{G})$ is a directed acyclic graph (DAG) with $V_\mathcal{G}$ a collection of nodes associated to random variables $\mathcal{X}$ (a node per variable), and $E_\mathcal{G}$ a collection of arcs; $\mathcal{P}$ is a collection of conditional mass functions $p(X_i|\Pi_i)$ (one for each instantiation of Π_i), where Π_i denotes the parents of X_i in the graph (Π_i may be empty), respecting the relations of $E_\mathcal{G}$. In a Bayesian network every variable is conditionally independent of its non-descendant non-parents given its parents (Markov condition). Because of the Markov condition, the Bayesian network represents a joint probability distribution by the expression $p(\mathbf{x}) = p(x_0, \ldots, x_n) = \prod_i p(x_i|\boldsymbol{\pi}_i)$, for every $\mathbf{x} \in \Omega_\mathcal{X}$ (space of joint configurations of variables), where every x_i and $\boldsymbol{\pi}_i$ are consistent with $\mathbf{x}$.

In the particular case of classification, the *class* variable X_0 has a special importance, as we are interested in its posterior probability which is used to predict unseen values; there are then several feature variables $\mathcal{Y} = \mathcal{X} \setminus \{X_0\}$. The supervised classification problem using probabilistic models is based on the computation of the posterior density, which can then be used to take decisions. The goal is to compute $p(X_0|\mathbf{y})$, that is, the posterior probability of the class variable given the values $\mathbf{y}$ of the features in a *test* instance. In this computation, p is defined by the model that has been learned from labeled data, that is, past data where class and features are all observed have been used to infer p. In order to do that, we are given a complete training data set $D = \{D_1, \ldots, D_N\}$ with N instances, where $D_u = \mathbf{x}_u \in \Omega_\mathcal{X}$ is an instantiation of all the variables, the first learning task is to find a DAG $\mathcal{G}$ that maximizes a given score function, that is, we look for $\mathcal{G}^* = \text{argmax}_{\mathcal{G}\in\boldsymbol{\mathcal{G}}} s_D(\mathcal{G})$, with $\boldsymbol{\mathcal{G}}$ an arbitrary set of DAGs with nodes $\mathcal{X}$, for a given score function s_D (the dependency on data is indicated by the subscript D).[1]

For the purpose of this work, we assume that the employed score is decomposable and respects likelihood equivalence. Decomposable means it can be written in terms of the local nodes of the graph, that is, $s_D(\mathcal{G}) = \sum_{i=0}^{n} s_D(X_i, \Pi_i)$. Likelihood equivalence means that if $\mathcal{G}_1 \neq \mathcal{G}_2$ are two arbitrary graphs over $\mathcal{X}$ such that both encode the very same conditional independences among variables, then s_D is likelihood equivalent if and only if $s_D(\mathcal{G}_1) = s_D(\mathcal{G}_2)$.

[1] In case of many optimal DAGs, then we assume to have no preference and argmax returns one of them.