Methods' details for the `bnclassify` package

Bojan Mihaljevic, Concha Bielza, Pedro Larranaga

2020-03-12

Abstract

This vignette provides details on the underlying methods and documents implementation specifics, especially where they differ from or are undocumented in the original paper. It complements the "overview" vignette.

Contents

1 Introduction

All notation and acronyms used here are introduced in `vignette("overview", package="bnclassify")`.

See the remaining vignettes:

- `vignette("overview", package="bnclassify")` provides details on the implemented methods.
- `?bnclassify` provides a a consise overview of the package, listing main functionalities and functions.
- `vignette("introduction", package="bnclassify")` provides details on the implemented methods.

2 Structure learning

2.1 Chow-Liu for one-dependence estimators

The CL-ODE algorithm by [Friedman et al., 1997] adapts the Chow-Liu [Chow and Liu, 1968] algorithm in order to find the maximum likelihood TAN model in time quadratic in n. Since the same method can be used to find ODE models which maximize decomposable penalized log-likelihood scores, `bnclassify` uses it to maximize Akaike's information criterion (AIC) [Akaike, 1974] and BIC [Schwarz, 1978]. While maximizing likelihood will always render a TAN, i.e., a network with $n-1$ augmenting arcs, maximizing penalized log-likelihood may render a FAN, since the inclusion of some arcs might degrade the penalized log-likelihood score.

Note that when data is incomplete **bnclassify** does not necessarily return the optimal (with respect to penalized log-likelihood) ODE. Namely, that requires the computationally expensive calculation of the sufficient statistics N_{ijk} which maximize parameter likelihood; instead, **bnclassify** approximates these statistics with the *available case analysis* heuristic (see Section 3).

2.2 TAN HC and TAN HC SP

TAN HC and TAN HC SP may evaluate equivalent structures at each step. Adding valid arcs $X_i \to X_j$ and $X_j \to X_i$ results in identical structures because of tree structure of the features subgraph. Namely, $|Pa(X) \setminus C| \leq 1$ for each X and thus we can only add the arc $X_i \to X_j$ if $Pa(X) = \{C\}$. Thus, adding an arc $X_i \to X_j$ introduces no v-structures into the network, and both $X_i \to X_j$ and $X_j \to X_i$ only remove the independence between X_i and X_j. The two obtained networks thus correspond to identical factorizations of the joint distribution.

To avoid scoring equivalent structures, at each step we selected the $X_i \to X_j$ such that X_i (that is, its column name in the data set) is alphabetically before X_j. A preferable implementation would be to select the arc randomly.

3 Parameter learning

3.1 Bayesian parameter estimation

bnclassify only handles discrete features. With fully observed data, it estimates the parameters with maximum likelihood or Bayesian estimation, according to Equation 2 in the "overview" vignette, with a single α for all local distributions. With incomplete data it uses *available case analysis* [Pigott, 2001] and substitutes $N_{\cdot j \cdot}$ in Equation 2 in the "overview" vignette with $N_{ij\cdot} = \sum_{k=1}^{r_i} N_{ijk}$, i.e., with the count of instances in which $\mathbf{Pa}(X_i) = j$ and X_i is observed:

$$\theta_{ijk} = \frac{N_{ijk} + \alpha}{N_{ij\cdot} + r_i \alpha}.$$

3.2 Exact model averaging for naive Bayes

The MANB parameter estimation method corresponds to exact Bayesian model averaging over the naive Bayes models obtained from all 2^n subsets of the n features, yet it is computed in time linear in n. The implementation in **bnclassify** follows the online appendix of Wei et al. [2011], extending it to the cases where $\alpha \neq 1$ in Equation~(3.1).

The estimate for a particular parameter θ_{ijk}^{MANB} is:

$$\theta_{ijk}^{MANB} = \theta_{ijk} P(\mathcal{G}_{C \not\perp\!\!\!\perp X_i} \mid \mathcal{D}) + \theta_{ik} P(\mathcal{G}_{C \perp\!\!\!\perp X_i}),$$

where $P(\mathcal{G}_{C \not\perp\!\!\!\perp X_i} \mid \mathcal{D})$ is the local posterior probability of an arc from C to X_i, whereas $P(\mathcal{G}_{C \perp\!\!\!\perp X_i}) = 1 - P(\mathcal{G}_{C \not\perp\!\!\!\perp X_i} \mid \mathcal{D})$ is that of the absence of such an arc (which is equivalent to omitting X_i from the model), while θ_{ijk} and θ_{ik} are the Bayesian parameter estimates obtained with Equation~(3.1) given the corresponding structures (i.e., with and without the arc from C to X_i).

Using Bayes' theorem,

$$P(\mathcal{G}_{C \not\perp\!\!\!\perp X_i} \mid \mathcal{D}) = \frac{P(\mathcal{G}_{C \not\perp\!\!\!\perp X_i}) P(\mathcal{D} \mid \mathcal{G}_{C \not\perp\!\!\!\perp X_i})}{P(\mathcal{G}_{C \not\perp\!\!\!\perp X_i}) P(\mathcal{D} \mid \mathcal{G}_{C \not\perp\!\!\!\perp X_i}) + P(\mathcal{G}_{C \perp\!\!\!\perp X_i}) P(\mathcal{D} \mid \mathcal{G}_{C \perp\!\!\!\perp X_i})}.$$

Assuming a Dirichlet prior with hyperparameter $\alpha = 1$ in Equation~3.1, Equation~(6) and Equation~(7) in the online appendix of Wei et al. [2011] give formulas for $P(\mathcal{D} \mid \mathcal{G}_{C \not\perp\!\!\!\perp X_i})$ and $P(\mathcal{D} \mid \mathcal{G}_{C \perp\!\!\!\perp X_i})$:

$$P(\mathcal{D} \mid \mathcal{G}_{C \not\perp\!\!\!\perp X_i}) = \prod_{j=1}^{r_C} \frac{(r_i - 1)!}{(N_{ij\cdot} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!,$$

$$P(\mathcal{D} \mid \mathcal{G}_{C \perp\!\!\!\perp X_i}) = \frac{(r_i - 1)!}{(N_i + r_i - 1)!} \prod_{k=1}^{r_i} N_{i \cdot k}!,$$

where $N_{i \cdot k} = \sum_{j=1}^{r_C} N_{ijk}$. Noting that the above are special cases of Equation~(8) in Dash and Cooper [2002], we can generalize this for any hyperparameter $\alpha > 0$ as follows:

$$P(\mathcal{D} \mid \mathcal{G}_{C \not\perp\!\!\!\perp X_i}) = \prod_{j=1}^{r_C} \frac{\Gamma(r_i \alpha)}{\Gamma(N_{ij} + r_i \alpha)} \prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk} + \alpha)}{\Gamma(\alpha)},$$

and

$$P(\mathcal{D} \mid \mathcal{G}_{C \not\perp\!\!\!\perp X_i}) = \frac{\Gamma(r_i \alpha)}{\Gamma(N_i + r_i \alpha)} \prod_{k=1}^{r_i} \frac{\Gamma(N_{ik} + \alpha)}{\Gamma(\alpha)}.$$

Following Wei et al. [2011], **bnclassify** asumes that the local prior probability of an arc from the class to a feature X_i, $P(\mathcal{G}_{C \not\perp\!\!\!\perp X_i})$, is given by the user. The prior of a naive Bayes structure $\mathcal{G}$, with arcs from the class to a out of n, features and no arcs to the remaining $n - a$ features is, then,

$$P(\mathcal{G}) = P(\mathcal{G}_{C \not\perp\!\!\!\perp X_i})^a (1 - P(\mathcal{G}_{C \not\perp\!\!\!\perp X_i}))^{(n-a)}. \tag{1}$$

Note that **bnclassify** computes the above in logarithmic space to reduce numerical errors.

3.3 Weighting to Alleviate the Naive Bayes Independence Assumption

The WANBIA [Zaidi et al., 2013] method updates naive Bayes' parameters with a single exponent 'weight' per feature. The weights are computed by optimizing either the conditional log-likelihood or the mean root squared error of the predictions. **bnclassify** implements the conditional log-likelihood optimization as described in the original paper, namely optimizing it with the L-BFGS [Zhu et al., 1997] algorithm, with its gradient $\mathbf{g}$ given by

$$g_i = \sum_{j=1}^{N} \Big(\log P(X_i = x_i^{(j)} \mid c^{(j)}) - \sum_{c \in C} P(c \mid \mathbf{x}; \mathbf{w}) \log P(X_i = x_i^{(j)} \mid c) \Big), \tag{2}$$

where the probabilities are those estimated with maximum likelihood, i.e., without taking weights into account, whereas $P(c \mid \mathbf{x}; \mathbf{w})$ takes weights into account. This corresponds to discriminative learning of parameters, as a discriminative, rather than generative, objective function is optimized.

If X_i is unobserved for some instance j, that is, $x_i^{(j)} = \texttt{NA}$, then we replace $P(X_i = x_i^{(j)} \mid c^{(j)})$ and $P(X_i = x_i^{(j)} \mid c)$ with 1 in Equation 2 (as a leaf in the Bayesian network, an unobserved X_i does not affect conditional log-likelihood).

3.4 Attribute-weighted naive Bayes

The AWNB parameter estimation method is intended for the naive Bayes but in **bnclassify** it can be applied to any model. It exponentiates the conditional probability of a predictor,

$$P(\mathbf{X}, C) \propto P(C) \prod_{i=1}^{n} P(X_i \mid \mathbf{Pa}(X_i))^{w_i},$$

reducing or maintaining its effect on the class posterior, since $w_i \in [0, 1]$ (note that a weight $w_i = 0$ omits X_i from the model, rendering it independent from the class.). This is equivalent to updating parameters of θ_{ijk} given by Equation~(3.1) as

$$\theta_{ijk}^{AWNB} = \frac{\theta_{ijk}^{w_i}}{\sum_{k=1}^{r_i} \theta_{ijk}^{w_i}},$$

and plugging those estimates into Equation 1 in the "overview" vignette. Weights w_i are computed as

$$w_i = \frac{1}{M} \sum_{t=1}^{M} \frac{1}{\sqrt{d_{ti}}},$$

where M is the number of bootstrap [Efron, 1979] subsamples from $\mathcal{D}$ and d_{ti} is the minimum testing depth of X_i in an unpruned classification tree learned from the t-th subsample ($d_{ti} = 0$ if X_i is omitted from t-th tree).

4 Prediction

`bnclassify` implements prediction for augmented naive Bayes models with complete data. This amounts to multiplying the corresponding entries in the local distributions and is done in logarithmic space, applying the *log-sum-exp* trick before normalizing, in order to reduce the chance of underflow.

With incomplete data this cannot be done and therefore **bnclassify** uses the `gRain` package [Højsgaard, 2012] to perform exact inference. Such inference is time-consuming and, therefore, wrapper algorithms can be very slow when applied on incomplete data sets.

References

Hirotugu Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

CK Chow and CN Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

Denver Dash and Gregory F Cooper. Exact model averaging with naive Bayesian classifiers. In *19th International Conference on Machine Learning (ICML-2002)*, pages 91–98, 2002.

Bradley Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1): 1–26, 1979.

N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29: 131–163, 1997.

Søren Højsgaard. Graphical independence networks with the `gRain` package for `R`. *Journal of Statistical Software*, 46(10):1–26, 2012.

Therese D Pigott. A review of methods for missing data. *Educational research and evaluation*, 7(4): 353–383, 2001.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

Wei Wei, Shyam Visweswaran, and Gregory F Cooper. The application of naive Bayes model averaging to predict Alzheimer's disease from genome-wide data. *Journal of the American Medical Informatics Association*, 18(4):370–375, 2011.

Nayyar A Zaidi, Jesus Cerquides, Mark J Carman, and Geoffrey I Webb. Alleviating naive Bayes attribute independence assumption by attribute weighting. *Journal of Machine Learning Research*, 14:1947–1988, 2013.

Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560, 1997.