Learning Bayesian Network Classifiers by Maximizing Conditional Likelihood

Daniel Grossman
Pedro Domingos
University of Washington
ICML 2004
Outline

• Bayesian Networks for Classification
• Generative vs. Discriminative training of the classifiers
• Bayesian Network structure search that optimizes conditional likelihood
• Experiments
• Conclusion
Background

- Naïve Bayes – special case of a BN is an accurate classifier, and outperforms BN (Friedman 1997)
- BN optimizes join-likelihood function, not class-conditional
- Class-conditional likelihood (CL) function optimization does not have simple closed form solution (Friedman)
- CL can be optimized by gradient ascent, but this may not be computationally feasible
Motivation for improving BN classifier

• Sometimes high accuracy is not enough, and we are interested in accurate class probabilities
  – Ranking of class probabilities
  – Cost-based classification
Learning BNs – scoring functions

• Need to learn structure and parameters
• Maximizing log-likelihood of the data

\[ LL(B|D) = \sum_{d=1}^{n} \log P_B(X_d) = \sum_{d=1}^{n} \sum_{i=1}^{v} \log P_B(x_{d,i}|\pi_{d,i}) \]

• Add complexity penalty: MDL minimizes

\[ MDL(B|D) = \frac{1}{2} m \log n - LL(S|D) \]

• Bayesian Score

\[ P(B_S, D) = P(B_S)P(D|B_S) \]

\[ = P(B_S) \prod_{i=1}^{v} \prod_{j=1}^{q_i} \frac{\Gamma(n_{ij}^l)}{\Gamma(n_{ij}^l + n_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(n_{ijk}^l + n_{ijk})}{\Gamma(n_{ijk}^l)} \]
BN Classifier

• Want to maximize probability of class given features (predictive attributes)

\[ P(y|x_1, \ldots, x_{v-1}) \]

• For classification purposes want to maximize class conditional log-likelihood instead of full log-likelihood

\[
CLL(B|D) = \sum_{d=1}^{n} \log P_B(y_d|x_{d,1}, \ldots, x_{d,v-1}).
\]

\[
LL(B|D) = CLL(B|D) + \sum_{d=1}^{n} \log P_B(x_{d,1}, \ldots, x_{d,v-1})
\]
Problem with maximizing LL

• Leads to underperformance of the classifiers since the contribution of the CLL will most likely be overruled by
  \[ \log P_B(x_{d,1}, \ldots, x_{d,v-1}) \]

• Want to maximize CLL directly, but cannot decompose
  \[ \sum_{d=1}^{n} \log [P_B(x_{d,1}, \ldots, x_{d,v-1}, y)/P_B(x_{d,1}, \ldots, x_{d,v-1})] \]

• Solution – gradient ascent
  – when the structure is known, can estimate parameters effectively (Greiner and Zhou, 2002)
  – when the structure is unknown, computationally infeasible, because need to compute gradient for each structure candidate
Discriminative vs. Generative models

To classify a new instance, want to know $P(Y|X)$

Discriminative models assume some functional form for $P(Y|X)$ and estimate parameters to maximize it from data

Generative models, assume probability distribution for data to estimate joint probability $P(X,Y)$ and compute $P(Y|X)$ by Bayes rule.

In practice with enough data discriminatively trained classifiers can significantly outperform generatively trained classifiers if the goal is classification accuracy.
BNC Algorithm

• Similar to hill-climbing proposed by Heckerman (1995), but uses CLL as an objective function
  – Start with empty network, at each step consider adding a new arc, and reversing/deleting each current arc without introducing cycles
  – Pre-discretizes continuous values
BNC Versions

- **BNC-nP**
  - To avoid over fitting, each variable is limited to $n$ parents. Parameters then would be set to their maximum likelihood (not CLL!) values. CLL is used to score the network.
  - Rationale – computing LL parameters is very fast, and for an optimal structure are asymptotically equivalent to maximum CLL

- **BNC-MDL**
  - Similar to BNC-nP, only uses scoring function
    \[
    CMDL(B|D) = \frac{1}{2}m \log n - CLL(S|D)
    \]
  - where $m$ is the number of parents, and $n$ is the size of the data.
Experiments

• Full optimization
  – Each parameters is set to its locally maximum CLL value by conjugate gradient. Parameters are initialized with likelihood function value. 2-fold cross validation used to prevent overfitting.
  • Speed-ups:
    – Only use 200 sampled for gradient, and entire data for fitting final parameters
    – Restrict the iterations gradient can take
  – Still takes a long time to run (1-2 hours small datasets, 5 hours medium dataset, on one dataset didn’t stop after 2 days)
Experiments

• 25 benchmark datasets from UCI Machine Learning repository, the same as those used by Friedman et. al.
• 5-fold cross-validation
• C4.5, Naïve Bayes, TAN, Hill-Climbing Greedy BN, Maximum Likelihood using MDL score, maximum likelihood restricting to 2 parents, NB and TAN with parameters optimized to CLL.
Results (error rate)
Future Work and Conclusion

• Future Work
  – Experimenting in the domain with large number of attributes
  – Developing heuristics for full optimization of CLL
  – Developing methods for handling over fitting
  – Handling missing data
  – Undiscretizing continuous variables
  – Extending their treatment to maximizing CLL on arbitrary query

• Conclusion
  – Presented classifier effectively searches for the structure that optimizes CLL producing good results