贝叶斯网络分类器近似学习算法①

郝宇晨

(山西财经大学 信息管理学院, 太原 030006)

摘 要: 贝叶斯网络在很多领域应用广泛, 作为分类器更是一种有效的常用分类方法, 它有着很高复杂度, 这使得贝叶斯网络分类器在应用中受到诸多限制. 通过对贝叶斯网络分类器算法的近似处理, 可以有效减少计算量, 并且得到令人满意的分类准确率. 通过分析一种将判别式算法变为产生式算法的近似方法, 介绍了这种算法的近似过程, 并将其应用在了贝叶斯网分类算法中. 接着对该算法进行分析, 利用该算法的稳定性特点, 提出 Bagging-aCLL 集成分类算法, 它进一步提高了该近似算法的分类精度. 最后通过实验确定了该算法在分类准确率上确有不错的表现.

关键词: 贝叶斯网络分类器; 产生式; 判别式; 近似算法; 集成

Bayesian Networks Classifier Using Approximation

HAO Yu-Chen

(Electronic Information Engineering, Shanxi University of Finance & Economics, Taiyuan 030006, China)

Abstract: Bayesian networks are widely used in many fields. As a classifier, it is an effective classification method. Bayesian network classifier is one of the most challenging problems, which makes the Bayesian network classifier subject to many limitations in the application. Through the pairs of Bayesian network classifier algorithms' approximate treatment, it can effectively reduce the amount of calculation, and get satisfactory classification accuracy. This paper analyzes a way to change discriminative score function to generative score function by approximation method. This way is applied in Bayesian network classification algorithm. Finally, this paper uses the stability of new algorithm, proposes a new classifier through integration called Bagging-aCLL. It uses ensemble to improve the accuracy rate of the algorithm. The experiment test shows the classification accuracy rate of the algorithm have a good performance.

Key words: Bayesian networks classifier;generative algorithm; discriminative algorithm; approximation; ensemble

贝叶斯网络分类器是基于贝叶斯网络[1]的分类模型, 该分类器具有计算高效、精确度高、具有坚实的理论基础等特点. 它可以很容易地从不同角度进行推广, 是数据挖掘、机器学习、模式识别中分类知识获取的重要工具之一. 贝叶斯网络分类算法是该分类器的基础. 贝叶斯网分类算法的目标是当给定一个完整的数据集合时, 通过评分函数的计算找到一个有向无环图, 使得当前图的评分函数最优, 从而最好的反映属性之间的相关程度来达到分类. 这个给定的评分函数, 分为产生式评分函数和判别式评分函数[2], 相应的贝叶斯网也就有产生式算法和判别式算法. 产生式算法通常是通过计算每个属性间的条件概率分布来进行计算, 会产生很多中间结果. 而判别式算法不需要得到很多的中间计算结果, 它将整个数据进行考虑, 它更看重分类器在实际中对实例进行分类的能力, 根据准确率来判断产生的分类器的优劣. 这两种算法都有各自的优缺点[3], 产生式算法速度较快但准确率一般, 判别式算法准确率较高但执行速度较慢.

由于贝叶斯网分类算法的复杂性, 有许多学者提出了将贝叶斯网分类算法进行近似的思想, 即在一定的分类准确率下对贝叶斯网算法进行近似处理. 这种近似可以由两种不同思路加以实现. 其一, 在简化搜索空间方面加以优化, 如 Ott 和 Miyano(2003)通过对贝叶斯网结构做出限制(节点的最大父节点数目)来降

① 收稿时间:2013-12-24;收到修改稿时间:2014-01-20

低贝叶斯网结构的复杂度; Perrier(2008)通过构造“超结构”来限制贝叶斯网结构的搜索空间; Kojima(2010)则提出了一种“父节点参数组”的新搜索策略来加快算法过程. 另一种思路则看重了优化评分函数, 如Kontkanen(1998)、Grossman (2004)在判别式评分函数LL上提出新的评分函数CLL; Silander(2010)也通过比较各种评分函数后提出新的近似的评分函数; Carvalho、Ross(2011)则提出了一种近似方法, 这种方法通过对判别式贝叶斯网分类器算法中的评分函数进行近似估计, 将判别式评分函数化为可以分解的产生式评分函数——aCLL.

本文对其中的 aCLL 算法进行分析, 并根据该算法弱稳定性的特点, 结合集成的思想对其进行改进, 提出 Bagging-aCLL 算法. 然后将其应用到了贝叶斯网分类器算法当中, 最后通过实验得出了 Bagging-aCLL 分类器算法在分类准确率上确实有较大的提高.

1　aCLL评分函数

贝叶斯网分类器节点定义为 $X=(X_1,\ldots X_n,C)$, 其中的 C 是类变量, 分类器的目的就是根据实例中节点的实际取值$(X_1,\ldots X_n)$来选择符合的类属性 C. 以 LL 评分函数为基础, 通过引入类变量, 就得到一个判别式评分函数: CLL[4].

$$CLL(B|D)=\sum_{t=1}^{N}\log P_B(c_t|y_t^1,\ldots,y_t^n)$$

由式子可见 CLL 评分函数是不能分解的, 必须整体计算, 这就阻止了将计算分配到每一个节点上. 判别式算法比产生式算法复杂的方面就在于此.

为了克服判别式评分函数不能分解的缺点, Carvalho、Ross 等人提出了一个近似过程[5], 即用另一个可分解的函数代替它. 假设类变量是二元的, C={0,1}. 则类属性的条件概率可以表达为:

$$P_B(c_t\mid y_t^1,\ldots,y_t^n)=\frac{P_B(y_t^1,\ldots,y_t^n,c_t)}{P_B(y_t^1,\ldots,y_t^n,c_t)+P_B(y_t^1,\ldots,y_t^n,1-c_t)}$$

为了简洁, 设 $U_t=P_B(y_t^1,\ldots,y_t^n,c_t)$, $V_t=P_B(y_t^1,\ldots,y_t^n,1-c_t)$, 则上述式子可写为:

$$P_B(c_t\mid y_t^1,\ldots,y_t^n)=\frac{U_t}{U_t+V_t}$$

更进一步, CLL 可以重新写为: CLL(B|D)=. $\sum_{t=1}^{N}[\log U_t-\log(U_t+V_t)]$ 设函数: $f(U_t,V_t)=\log\frac{U_t}{U_t+V_t}$, 下面通过对这个函数的近似来估计 CLL 函数. 设

$$\hat{f}(U_t,V_t)=\alpha\log U_t+\beta\log V_t+\gamma$$

其中 α, β, γ 都是实数. 通过选取适当的 α, β, γ 值, 使 $\hat{f}(U_t,V_t)$ 函数尽可能的接近 $f(U_t,V_t)$ 函数. 当然, 实际中用多项式函数去估计对数函数是不可能的, 这就需要做出一些假设[5], 根据这些假设, 多项式函数就能在一定程度上对对数函数进行近似.

这个近似过程将通过最小均方差来实现, 通过计算[6], 得出准确的参数值:

$$\alpha=\frac{\pi^2+6}{24},\ \beta=\frac{\pi^2-18}{24},\ \gamma=\frac{\pi^2}{12\ln 2}-(2+\frac{(\pi^2-6)\log p}{12})$$

其中 p 为一个接近 0 的实数. 并且通过计算, p 并不能改变 $\hat{f}(U_t,V_t)$ 和 $f(U_t,V_t)$ 之间的差. 使用这个假设, 用 $f(U_t,V_t)$ 来估计 CLL, 得到:

$$CLL(B|D)\approx(\alpha+\beta)LL(B\mid D)-\beta\sum_{t=1}^{N}\log(\frac{U_t}{V_t})+N\gamma$$

其中 α, β, γ 是常数. 为了使 CLL 评分函数最大, 则最后一项 $N\gamma$ 作为一个固定的实数就可以忽略. 所以, aCLL 评分函数最后化为:

$$aCLL(B\mid D)=(\alpha+\beta)LL(B\mid D)-\beta\sum_{t=1}^{N}\log(\frac{U_t}{V_t})$$

这样, 就得到了一个可以分解的、近似的条件对数评分函数. 这种可分解的性质大大降低了算法在计算过程中的复杂性, 通过这个转换, 就将判别式评分函数近似的转换为产生式评分函数, aCLL 评分函数不仅有着判别式的高准确率, 又有着产生式评分函数的高效率, 将它应用在贝叶斯网分类算法中不失为一种很好的选择.

2　贝叶斯集成分类器Bagging-aCLL

如前所述, aCLL 评分函数在分类问题上取得了高准确率与高效率的平衡, 虽然近似过程包含着假设条件, 但通过实验可知这种假设是可以接受的, 有着很好的实际效果. 但包含假设条件的 aCLL 评分函数也有自身的特点, 即使用 aCLL 评分函数的贝叶斯网分类算法稳定性较弱. 学习算法的稳定性常用来衡量算法是否对训练集有很大的关联. 如果训练集有较小的

变化, 学习算法产生的预测函数发生较大变化, 则这个学习算法可以被认为是不稳定的. 由上述理论可知, 算法是将贝叶斯网络上的每个节点及其父节点情况的评分做和来得出最后的评分, 如果给定的数据集不同, 单个节点的评分不同, 最后得到的评分必然有些许的差别. 虽然 aCLL 评分函数对原有判别式算法有所改进, 但最终结果很可能是对训练集学习后的贝叶斯网络, 有着过度拟合现象, 并不能很好的对整个模型进行预测, 这就要通过一定的手段使该算法的分类精度进一步得到提高, 其中应用集成就是一种很好的思路.

集成有多种方法, 本文选择 Bagging[7]的算法思想作为 aCLL 算法的集成方式来提高它的分类精度. 除了上述由于算法实现过程产生的不稳定性外, 在具体的算法执行过程中, 还有实现算法时从哪个节点开始计算也会成为算法不稳定的原因. 例如本文在实现 aCLL 评分函数时, 采用的结构搜索策略为常用的爬山法搜索策略. 而在进行爬山法的每一轮的学习过程中, 搜索策略会从确定的节点开始进行, 使得最后得到的模型在很大程度上与最先选择的节点相关. 训练集在这个节点上的数据对最终的模型产生有着很大的作用, 为了消除单个节点对算法的影响, 除了要考虑多种不同的训练集, 更好的做法是在每一轮训练模型时, 爬山法的起始节点应该不同. 为了进一步提高 aCLL 贝叶斯网分类算法的分类精度, 本文利用集成算法的思想, 对 aCLL 算法进行集成, 提出了 Bagging-aCLL 算法. 该算法的基本思路是让 aCLL 算法训练多轮, 每一轮都采用不同的、随机的训练集, 每轮的训练集由从初始的训练集中随机取出的训练例组成, 初始训练例在某轮训练集中可以出现多次或根本不出现. 训练之后可得到一个预测模型序列, 最终的预测模型采用投票方式来做出选择. 并且在爬山法搜索策略上, 每一轮都随机选择一个不同的节点作为初始节点. 只有在最大程度上增加算法在训练集上的随机计算性, 才能消除训练结果的过度拟合现象. 从而提高了 aCLL 算法的准确率. 该算法的算法流程如图 1 所示.

Bagging-aCLL 算法在做集成时, 通过对爬山法起始节点的随机设置, 就能在每一轮得到起始节点不同的贝叶斯网结构. 这种做法消除了特殊节点对最优贝叶斯网结构的影响, 而集成的思路又消除了特定训练集对贝叶斯网结构的影响. 这样, 即使训练集做出稍微改变, 都能更准确的反应实际的分类模型.

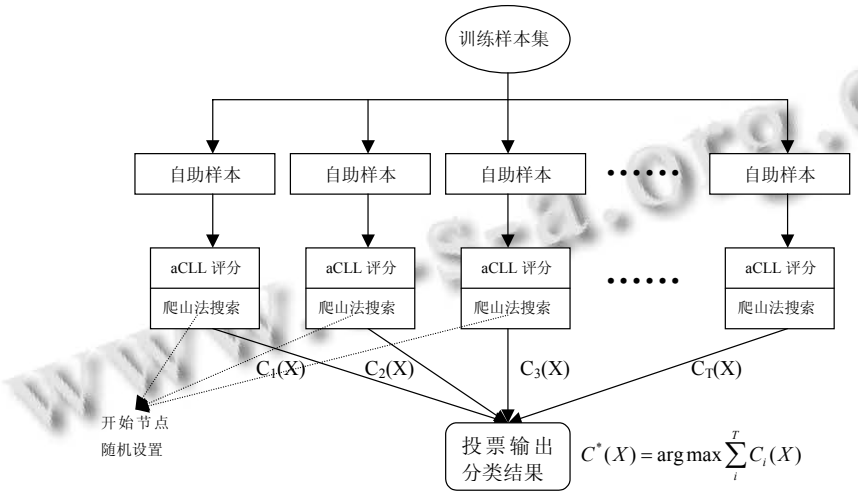

图 1 Bagging-aCLL 集成分类器算法流程

3 实验结果及分析

本文实验的平台使用了在数据挖掘方面应用广泛的 Weka 平台. 本实验的硬件环境如下. CPU: Intel(R) Pentium(R) 3.0GHz×2; 内存: 1GB; 硬盘: 120GB; 操作系统: windows XP sp2.

实验数据来自标准 UCI 仓库(Newman, 1998)[8], 共有 15 组数据. 在数据预处理时, 连续值属性都会先离散化, 并且将丢失值从数据集中去除, 然后转化为 Weka 所识别的 arff 格式文件. 由于 Bagging-aCLL 算法仅能处理类个数为 2 的属性集, 所以选取的属性集

当中类个数均为 2. 在选取的实验数据集中, 既有实例个数较少的数据集, 如 hepatitis、cleve、corral, 也有实例个数较多的数据集, 如 chess、letter, 这些数据集包含了众多领域, 具有很好的代表性, 根据它们能够很好的分析出 Bagging-aCLL 算法与产生式、判别式算法的差异, 同时可以直观表现出 Bagging-aCLL 算法对 aCLL 算法准确率的提升效果. 它们的属性个数及实例个数如表 1 所示.

表 1　训练数据的特征

	数据集	属性个数	实例个数
1	australian	15	690
2	breast	10	683
3	chess	37	2130
4	cleve	14	128
5	corral	7	128
6	crx	16	635
7	diabetes	9	768
8	flare	11	1066
9	german	21	1000
10	glass	10	163
11	heart	14	270
12	hepatitis	20	80
13	iris	5	150
14	letter	17	15000
15	mofn	11	300

为了确认 aCLL 算法是否在准确率方面要优于传统评分函数, 实验首先对 aCLL 算法分类器与传统的产生式评分函数 Bayes 分类器和判别式算法 CLL 分类器进行了分类准确率比较, 然后又与 Bagging-aCLL 分类器算法进行比较, 看是否可以提高分类的稳定性. 通过多次实验取平均值(括号内为标准差), 得到的实验结果如表 2 所示, 其中准确率较大的数据加粗处理以直观表现.

由实验可知, aCLL 分类器在准确率方面与判别式算法 CLL 分类器相当, 要优于产生式算法 Bayes 分类器. 实验中 aCLL 算法在时间复杂度方面介于二者之间. 正是因为将判别式算法进行了某种近似, 使得判别式算法的评分函数可以被分解, 而用产生式的算法来进行计算, 这样就综合了两种评分函数的优点. 这样, aCLL 算法既有了判别式算法的高分类准确率, 又有了产生式算法的较低时间复杂性.

表 2　Bagging-aCLL 算法与其他算法的比较

分类器	aCLL	Bayes	CLL	Bagging-aCLL
数据集	准确率(%)	准确率(%)	准确率(%)	准确率(%)
australian	85.51(1.70)	84.26(1.29)	85.22(2.25)	**86.57(0.52)**
breast	97.66(0.45)	96.46(1.10)	96.19(1.11)	**97.67(0.20)**
chess	96.90(0.00)	96.90(0.00)	96.90(0.00)	**97.65(0.99)**
cleve	84.12(3.25)	80.46(4.22)	**85.42(4.90)**	85.42(1.24)
corral	**99.22(0.00)**	**99.22(0.00)**	**99.22(0.00)**	**99.22(0.00)**
crx	87.14 (3.46)	86.85(4.53)	86.99(3.89)	**87.85(3.40)**
diabetes	78.91(9.24)	**79.29(10.24)**	78.91(10.29)	79.25(9.26)
flare	82.55(2.36)	82.16(2.37)	82.74(2.67)	**83.31(2.12)**
german	74.20(7.21)	72.29(7.63)	**75.02(6.97)**	**75.02(6.88)**
glass	85.89(0.00)	85.89(0.00)	85.89(0.00)	**85.96(0.05)**
heart	83.70(2.25)	**85.93(2.12)**	82.22 (2.33)	83.88(1.05)
hepatitis	90.00(3.35)	85.00(3.99)	88.75(3.53)	**91.05(3.16)**
iris	**94.00(1.94)**	**94.00(1.94)**	**94.00(1.94)**	**94.00(1.94)**
letter	**86.40(0.48)**	86.06(0.49)	86.14(0.49)	86.10(0.49)
mofn	90.90(1.66)	90.04(1.73)	**90.61(1.68)**	89.06(1.05)

进一步, 在实际应用中, 利用 aCLL 算法稳定性特点, 通过集成来提高 aCLL 算法的准确率, 实验中通过使用 Bagging-aCLL 算法分类器, 在每一轮中都用爬山法重新学习一个起始节点不同的贝叶斯网络结构, 最后用投票方式选择最优的结构. 由于采用集成的方法, 对多个分类器进行计算势必会增加时间复杂 度, 但与其它分类器的集成算法相比较, 时间复杂度相差不大, 如果采用并行计算的方法, 使单个分类器同时进行计算, 则时间复杂度会大大降低. 而通过计算它的分类准确率, 可以看出分类准确率确实有一定的提高, 这样的集成算法利用了 aCLL 算法的特点, 进一步提高了分类精度. 可见 Bagging-aCLL 分类器可以在准确率上取得令人满意的效果.

4　结语

本文分析了一种将判别式评分函数进行近似的方法, 通过这种近似, 使判别式可以被分解. 分解后的判别式评分函数的计算化为对每个节点评分的计算再做和, 大大化简了以前需要整体计算的过程, 在保持分类准确率较高的情况下减少了时间复杂度. 本文得到的 Bagging-aCLL 算法就是应用这种近似过程得到的近似贝叶斯网分类算法. 它不仅含有两种分类方法各自的优点, 还通过集成思想对其分类精度做了进一

步的提升, 从实验当中得出 Bagging-aCLL 算法是一种很好的贝叶斯网络分类器算法.

参考文献

1 张连文,郭海鹏.贝叶斯网引论.北京:科学出版社, 2006:61−64

2 Carvalho M. Scoring function for learning Bayesian networks[Technical Report], Inesc-id Tec. 2009.

3 Bouchard G, Triggs B. The trade off between generative and discriminative classifier. Proc. of COMPSTAT'04. Prague. Springer. 2004. 721−728.

4 Grossman D, Domingos P. Learing Bayesian network classifiers by maximizing conditional likelihood. Proc. of the twenty-first international conference on Machine learning. ACM. 2004. 46.

5 Carvalho AM, et al. Discriminative learning of bayesian networks via factorized conditional log-likelihood. Journal of Machine Learning Research, 2011, 12: 2181−2210.

6 Carvalho M, Oliveira AL, Sagot MF. Efficient learning of Bayesian network classifiers. Proc. IA'07. 2007.

7 Breiman L. Bagging predictors. Machine Learning, 1996, 24: 123−140.

8 Newman J, Hettich S, Blake CL, Merz CJ. UCI repository of machine learning databases, 1988. URL http://www.ics.uci.edu/~mlearn/MLRepository.html