软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2017,28(11):2891−2904 [doi: 10.13328/j.cnki.jos.005350]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于显露模式的数据流贝叶斯分类算法*

杜 超, 王志海, 江晶晶, 孙艳歌

(北京交通大学 计算机与信息技术学院,北京 100044)

通讯作者: 王志海, E-mail: zhhwang@bjtu.edu.cn

摘 要: 基于模式的贝叶斯分类模型是解决数据挖掘领域分类问题的一种有效方法.然而,大多数基于模式的贝叶斯分类器只考虑模式在目标类数据集中的支持度,而忽略了模式在对立类数据集合中的支持度.此外,对于高速动态变化的无限数据流环境,在静态数据集下的基于模式的贝叶斯分类器就不能适用.为了解决这些问题,提出了基于显露模式的数据流贝叶斯分类模型 EPDS(Bayesian classifier algorithm based on emerging pattern for data stream).该模型使用一个简单的混合森林结构来维护内存中事务的项集,并采用一种快速的模式抽取机制来提高算法速度.EPDS 采用半懒惰式学习策略持续更新显露模式,并为待分类事务在每个类下建立局部分类模型.大量实验结果表明,该算法比其他数据流分类模型有较高的准确度.

关键词: 数据流;显露模式;贝叶斯;数据挖掘
中图法分类号: TP181

中文引用格式: 杜超,王志海,江晶晶,孙艳歌.基于显露模式的数据流贝叶斯分类算法.软件学报,2017,28(11):2891−2904. http://www.jos.org.cn/1000-9825/5350.htm

英文引用格式: Du C, Wang ZH, Jiang JJ, Sun YG. Bayesian classifier algorithm based on emerging pattern for data stream. Ruan Jian Xue Bao/Journal of Software, 2017,28(11):2891−2904 (in Chinese). http://www.jos.org.cn/1000-9825/5350.htm

Bayesian Classifier Algorithm Based on Emerging Pattern for Data Stream

DU Chao, WANG Zhi-Hai, JIANG Jing-Jing, SUN Yan-Ge

(School of Computer Science and Technology, Beijing Jiaotong University, Beijing 100044, China)

Abstract: Pattern-Based Bayesian model is one of the solutions for the classification problem in data mining. Most pattern-based Bayesian classifiers consider the supports of patterns in the dataset of the home class only. However, the supports of the patterns in the counterpart class are ignored. In addition, for the high-speed dynamic changes and infinite data stream, pattern-based Bayesian classifier which aims at static datasets can not work. To overcome these problems, EPDS (Bayesian classifier algorithm based on emerging pattern for data stream) is proposed. EPDS is a Bayesian classification model based on the emerging patterns discovered over data stream. In this model, EPDS presents a simple hybrid forests (HYF) data structure to maintain the itemsets of the transactions in memory, and uses a fast pattern extracting mechanism to accelerate the algorithm. EPDS adopts partially-lazy learning strategy to update emerging itemsets continuously, and establishes a local classification model in each class for the test transaction. Experimental results on real and synthetic data streams show that EPDS achieves higher classification accuracy compared to other classic classifiers.

Key words: data stream; emerging pattern; Bayesian; data mining

数据流中数据以高速的、无限流动的形式产生[1],并且数据的分布可能随着时间发生变化.相对于传统的静态数据,基于数据流的模式分类模型要面对一些新的挑战[2]:(1) 算法只能对数据进行一次扫描(one-pass);

* 基金项目: 国家自然科学基金(61672086)
Foundation item: National Natural Science Foundation of China (61672086)
本文由复杂环境下的机器学习研究专刊特约编辑胡清华教授、张道强教授、张长水教授推荐.
收稿时间: 2017-05-15; 修改时间: 2017-06-16, 2017-08-23; 采用时间: 2017-09-06

(2) 算法要能适应数据的变化;(3) 算法要受到计算机内存空间的限制.

在传统的静态数据集合环境下,基于模式的贝叶斯分类模型是解决分类问题可行的办法之一,指的是在大规模数据集合中抽取模式集合,再根据抽取到的模式集合建立贝叶斯分类模型.然而,在静态数据集合中的算法不能满足数据流环境的挑战,且基于模式的贝叶斯分类模型大多关注于模式 X 在待预测类 c_i 的数据集合(即目标数据集合)中的支持度,而忽略了对模式 X 在其他非 c_i 类的数据集合(即对立数据集合)中支持度的考量,也就是忽略了对模式 X 在 c_i 类和非 c_i 类之间区分度的考量,进而影响分类效果.

为此,本文提出了基于显露模式(emerging pattern,简称 EP)[3]的数据流贝叶斯分类模型(Bayesian classifier algorithm based on emerging pattern for data stream,简称 EPDS).显露模式是支持度从一个数据集到另一个数据集有明显变化的模式[4,5].显露模式能够捕捉两类数据集上属性之间的差异,具有很好的分类性能[4].

EPDS 使用半懒惰式学习策略.半懒惰式学习(partially-lazy learning)是一种介于懒惰式学习和急切式学习之间的学习策略.半懒惰式分类器适于处理高度动态和复杂的学习环境,例如数据流[6].EPDS 分类模型在训练阶段用数据流的滑动窗口机制捕获局部数据,并随数据的流动实时更新局部数据.当有分类请求时,EPDS 根据待分类事务从当前局部数据中抽取显露模式,并建立针对待分类事务的特定局部分类模型.

为了满足数据流环境下挖掘显露模式的要求,本文提出了一种单次扫描算法(find emerging pattern algorithm on data stream,简称 FEP).为了实现对数据的密集存储,该算法基于 SFI-forest(summary frequent itemset forest)[7]提出了结构更为简单的混合森林(hybrid-forest,简称 HYF)结构来存储捕获到的数据所包含的项集.同时提出了基于待分类事务的显露模式抽取机制以提高算法效率.

EPDS 分类模型抽取相互独立的,既能表达尽可能多的属性之间的关联关系,又具有尽可能高的类间区分度的显露模式.对数据流中显露模式抽取不完全的情况和显露模式增长率很高而支持度很低的情况,EPDS 用平滑技术处理未被覆盖的项,并用增长率对显露模式的支持度进行修正.

本文的创新点如下:

(1) 采用半懒惰式学习策略,在不断更新的显露模式集合上,根据待分类事务建立局部的分类模型,提高分类效率;
(2) 提出了能够满足数据流分类任务需要的简单的混合森林结构来对数据集中的项进行密集存储,提升数据流的处理速度;
(3) 提出了给定项限制的显露模式抽取机制,减少生成的候选项集数量,并在显露模式抽取不完全的情况下对未被抽取的项使用平滑技术处理.

本文第 1 节介绍本文用到的基本概念和相关的研究现状.第 2 节描述数据流上显露模式的挖掘算法,具体包括基于滑动窗口的混合森林结构的建立和更新机制以及显露模式的挖掘方法.第 3 节介绍 EPDS 分类器,具体包括选取显露模式建立贝叶斯概率估计的原则、修正显露模式支持度的方法以及分类器的训练和预测过程.第 4 节将通过实验来评价 EPDS 分类器.第 5 节讲述本文的结论和将来的工作.

1 相关概念及研究现状

基于数据流显露模式表示的贝叶斯分类器利用挖掘显露模式所发现的属性之间的相互关系来计算对贝叶斯联合概率的估计.本节将介绍相关的基本概念和研究现状.

1.1 基本概念和定义

本文关注数据流环境下显露模式挖掘及其分类.相关的定义如下.

定义 **1**(数据流). 事务数据流 $DS=[T_1,T_2,\ldots,T_n,\ldots)$是事务的无限序列.其中,$T_n$ 表示当前最新的事务,是数据流 DS 的第 n 个事务.

定义 **2**(项). 设 A 是数据的一个标签,常被称为属性,用于描述数据的一个特征.Ω是属性 A 对应的离散的值域,那么项是属性 A 与其取值 a_i(其中,$a_i\in\Omega$)的序偶$\langle A,a_i\rangle$.通常用属性取值 a_i 来表示项$\langle A,a_i\rangle$.

设 $\boldsymbol{A}=\{A_1,A_2,\ldots,A_n\}$表示属性的集合,$\boldsymbol{\Omega}=\{\Omega_1,\Omega_2,\ldots,\Omega_n\}$表示 n 个离散型属性分别对应的取值空间.数据流 DS

的一个事务 $T_{tid}=\{a_1,a_2,\ldots,a_m\}$是项的集合,$tid$ 是事务的标识,m 是事务 T_{tid} 中项的个数,$m\leq n$.每个事务 T_{tid} 中只包含每个属性的最多 1 个项.

定义 **3**(数据流项集的支持度). 数据流 DS 和项集 X,X 的支持度($Support(X)$)是指数据流 DS 中包含项集 X 的事务的数量($Count(X)$)与数据流 DS 所有事务的数量($|DS|$)的商,如公式(1)所示.

$$Support(X)=\frac{Count(X)}{|DS|} \tag{1}$$

因为数据流 DS 是无限的事务集合,所以直接用数据流中所有事务的数量来计算支持度是不合适的.本文用滑动窗口模型进行数据流的挖掘任务.

定义 **4**(滑动窗口). 滑动窗口 $W_{n-|W|+1}=[T_{n-|W|+1},T_{n-|W|+2},\ldots,T_n]$是定义在数据流 DS 上包含最近的$|W|$个事务的集合,其中,$|W|$是滑动窗口的大小,也就是窗口中事务的个数.

在滑动窗口机制中,我们只考虑距离当前最近的$|W|$个事务.此时,项集 X 的支持度可以表示为

$$Support(X)=\frac{Count(X)}{|W|} \tag{2}$$

因为本文要研究数据流的分类问题,不光需要知道项集在窗口所有数据集中的支持度,还需要知道项集在由各个不同类的事务所构成的类数据集中的支持度,我们称其为数据流项集的类支持度.

定义 **5**(数据流项集的类支持度). 设 C 是数据流 DS 的类属性,$\Omega_C=\{c_1,c_2,\ldots,c_k\}$是类属性的取值空间,$C_i$ 表示窗口 W 中类属性值为 c_i 的事务的集合,那么数据流项集 X 的类支持度($Support_i(X)$)的定义为

$$Support_i(X)=\frac{Count(X)|_{C=c_i}}{|C_i|} \tag{3}$$

其中,$Count(X)|_{C=c_i}$ 是当前窗口的事务集合 C_i 中包含项集 X 的事务的数量.

定义 **6**(频繁模式). 数据流 DS 的窗口 W 中,当项集 X 的支持度 $Support(X)$大于最小支持度阈值($minSupport$)时,则称 X 为频繁项集,也称为频繁模式.频繁模式的集合 $FP=\{X|Support(X)\geq minSupport\}$.

定义 **7**(项集的增长率)[3]. 数据流 DS 的窗口 W 中,所有事务被划分为两个不同的事务子集 D_1 和 D_2,即 $D_1\cap D_2=\varnothing$,$D_1\cup D_2=D$,项集 X 从 D_1 到 D_2 的增长率 $Growth(X,D_1,D_2)$定义如下:

$$Growth(X,\boldsymbol{D}_1,\boldsymbol{D}_2)=\begin{cases}0, & Support_{D_1}(X)=Support_{D_2}(X)=0\\ \infty, & Support_{D_2}(X)>0,Support_{D_1}(X)=0\\ \dfrac{Support_{D2}(X)}{Support_{D1}(X)}, & \text{其他}\end{cases} \tag{4}$$

定义 **8**(显露模式). 项集 X,如果 $Growth(X,D,D_2)$不小于给定的增长率阈值ρ,即 $Growth(X,D_1,D_2)\geq\rho$,就可以称 X 为从事务集合 D_1 到事务集合 D_2 的ρ-EP,或者简单地称项集 X 是 D_2 的 EP.其中,D_2 称为显露模式的目标事务集合,D_1 称为显露模式的对立事务集合.

我们把类值为 c_i 的事务集合称为目标事务集,记为 C_i;所有非 c_i 类的事务集合称为对立事务集合,记为 C_i'.项集 X 从对立事务集 C_i' 到目标事务集 C_i 的增长率就表示 $Growth(X,C_i',C_i)$.本文就是要从窗口事务中为每个类 c_i 挖掘 $Growth(X,C_i',C_i)\geq\rho$的显露模式,并基于显露模式建立贝叶斯分类模型.

1.2 研究现状

分类是指在已有的训练数据上学习一个分类模型,以建立数据集中的数据记录与给定类别之间的映射关系,从而可以对类别未知的数据记录进行预测.贝叶斯分类器[8]是一种已被广泛研究的分类器.对贝叶斯理论中联合概率的计算,是构造贝叶斯分类模型的一个难点.研究者往往通过一些简化模型来降低联合概率的计算难度,其中,最经典的简化模型是 NB(naïve Bayes),这是一个条件独立性假设模型.NB 模型把概率 $P(T_{test},c_i)$的计算转化为公式(5)的形式.

$$P(T_{test},c_i)=P(a_1,a_2,...,a_n,c_i)\approx P(c_i)P(a_1\mid c_i)P(a_2\mid c_i)...P(a_n\mid c_i)=P(c_i)\prod_{j=1}^{n}P(a_j\mid c_i) \quad (5)$$

然而在现实数据中,NB 提出的这种条件独立性假设是很少成立的.因此,之后又提出了许多弱化 NB 的强条件独立性假设的算法.总的来说,这些算法可以分为两种类型:一种是贝叶斯网络[1,9],这种算法研究的是属性之间的低阶依赖;另一种是基于模式的贝叶斯算法,这种算法研究的是属性之间的高阶依赖,通过在数据集中抽取频繁模式来建立联合概率 $P(T_{test},c_i)$的乘积近似值[10−13].目前,存在条件依赖模型和条件独立模型两种估计联合概率乘积近似值的模型.

条件依赖模型给出了属性间的依赖关系.设给定的数据流 DS 包含属性 A_1,A_2,A_3,A_4,A_5 和类属性 $C.c_i$ 是类属性的任意值,$T_{test}=\{a_1,a_2,\ldots,a_5\}$是待分类事务,公式(6)是条件依赖模型对联合概率 $P(T_{test},c_i)$估计的表示形式:

$$P(T_{test},c_i)=P(a_1,a_2,\ldots,a_5,c_i)\approx P(c_i)P(a_1|c_i)P(a_2|a_1c_i)\ldots P(a_5|a_1a_2a_3a_4c_i) \quad (6)$$

条件独立模型给出了属性集间的独立关系.公式(7)是条件独立模型对联合概率 $P(T_{test},c_i)$估计的表示形式:

$$P(T_{test},c_i)=P(a_1,a_2,\ldots,a_5,c_i)\approx P(c_i)P(a_1a_2|c_i)P(a_3a_4a_5|c_i) \quad (7)$$

在已有的基于静态数据集合的贝叶斯估计方法中,一类是基于条件依赖模型构建乘积近似估计[9,14].当有分类请求时,模型抽取相应的频繁项集,并依据条件依赖模型用抽取的项集建立乘积近似估计.这种方法为待分类事务在每个类下都建立相同结构的乘积近似估计.另一类是基于条件独立模型来建立联合概率的乘积近似估计[10,11].EnBay[10]分类算法选择相互独立的、尽可能长的、能够完全覆盖待分类事务的、尽可能少的频繁项集来建立分类器.EnBay 在各类标值下为待分类事务构建不同结构的贝叶斯联合概率的乘积近似估计.但是 EnBay 需要反复计算项集所属属性集之间的依赖程度,影响了响应分类请求的速度.

DSM-FI[7]算法用于在事务型数据流上增量地挖掘频繁项集.算法是单次扫描的批处理算法,使用界标窗口模型来获取流数据,提出了 SFI-forest 数据结构用于保存数据流中事务的摘要信息,并且提出了频繁项集查找机制.然而,该算法更关注挖掘长频繁模式.MSW[15]算法采用 SW-tree 单遍扫描流数据并捕获最新的模式信息,进而挖掘数据流任意大小滑动窗口内的频繁模式.然而 SW-tree 中各结点需按预先定义的全序关系排列,影响算法的通用性.且两个算法都没有进行分类的挖掘.

显露模式能够捕获带时间戳数据的显著趋势,或不同类型数据之间的显著特征[5].当前,显露模式主要用于解决分类问题[16].基于显露模式的分类器有分类精度高、易于理解等特点[17],已被应用于流数据分析、入侵检测、图像处理等领域[5,17,18].BCEP[19]和 MaxEPs[20]是两个著名的基于显露模式的贝叶斯分类模型.BCEP 算法选择长的和支持度大的显露模式来估计每个类的贝叶斯乘积近似.MaxEPS 算法选择没有任何超集是显露模式的最长显露模式来构建贝叶斯分类模型.两种算法都通过缩小候选项集的范围来提高计算效率.但这两种算法都存在构建分类器的显露模式之间有重叠的问题,且都没有用于数据流环境.DFP-SEPSF[16]采用动态频繁模式树结构实现对项集的压缩存储,实现在数据流环境下挖掘频繁模式,并且提出了不同于文献[3]中边界运算的挖掘强显露模式的新方法,但是并没有提出基于显露模式构建分类器的方法.

2 基于数据流的显露模式挖掘算法

本文提出了一种单次扫描算法 FEP,采用滑动窗口机制获取数据流的数据,并采用结构更为简单的混合森林结构存储数据中的项,进而挖掘频繁项集,然后利用边界运算挖掘显露模式.

2.1 数据流中项集的生成和更新

FEP 算法使用滑动窗口机制来获取数据流中的数据.设当前滑动窗口 $SW=[T_1,T_2,\ldots,T_{|SW|}]$,其中,$|SW|$是滑动窗口的大小,$T_i$为 FEP 读取的事务,$i\in\{1,2,...,|SW|\}$.将 T_i 按类标 $c_i\in C$(设定类属性的取值空间为 C)进行划分,在划分的事务集上建立子森林 HYF_i,由各个划分的事务集合上建立的子森林结构组成混合森林结构,$HYF=\{HYF_i\}$,$\forall c_i\in C$.根据待分类事务抽取项集时,将分别在不同类值的子森林结构 HYF_i 中抽取相应的项集.算法主要包括以下 3 个步骤.

- 首先,FEP 算法读取当前窗口中的事务,并且按其类标值对事务进行划分,然后将事务插入到相应划分

的子森林结构 HYF_i,其中,$c_i \in C$ 是读取的当前事务的类标值;

- 然后,当窗口中的事务更新时(删除旧事务,接收新事务),FEP 算法从混合森林结构 HYF 的子森林结构 HYF_j(HYF_j 是被删除的旧事务所在的划分的子森林结构)中删除旧事务的信息,并把当前窗口新接收的事务的信息添加到相应类标值的子森林结构中;
- 最后,当接到分类请求时,FEP 算法结合待分类事务中的项的限制,在依据当前窗口事务集合所建立的混合森林结构 $HYF=\{HYF_i\}$的各子森林结构中分别抽取每个类的频繁项集集合,再利用抽取的频繁项集挖掘每个类的显露模式.

混合森林结构 HYF 由多个子森林结构组成,每个子森林结构是对数据流 DS 中任意类 c_i 的所有事务的紧凑表示,标记为 HYF_i.混合结构 HYF 的定义如下.

(1) $HYF=\{HYF_i\}, i\in\{1,2,\ldots,k\}$.每个子森林结构 HYF_i 都包含 3 个组成部分:频繁项列表(a list of frequent items,简称 FI-list)、频繁项树(frequent item tree,简称 FIT)和辅助频繁项列表(a list of subsequence frequent items,简称 SFI-list).

(2) FI-list 是当前窗口中类 c_i(其中,$i\in\{1,2,\ldots,k\}$)的事务集中包含的所有项的列表.FI-list 中的每一个元素都由 3 个部分组成:$\{x,x.count,x.head\text{-}link\}$,其中,$x$ 表示项,$x.count$ 表示当前子混合森林结构中包含项 x 的事务个数.$x.head\text{-}link$ 用于指向以该项为根的 x.FIT 中的根结点,对于 FI-list 中的每一个项,都对应有一个 x.FIT 和一个 x.SFI-list.

(3) x.FIT 是以项 x 为根的树,树中每一个结点都由 3 个部分组成:$\{x,x.count,x.node\text{-}link\}$,其中,$x.count$ 表示包含该结点的所有事务的个数.$x.node\text{-}link$ 用于指向分支中的下一个结点,一般为项 x 所在事务的后续项.

(4) x.SFI-list 是辅助项 x 抽取频繁项集的列表,保存以项 x 为根的 x.FIT 中包含的所有项的信息.x.SFI-list 中的每个元素有两个部分组成:$\{e,e.count\}$,其中,e 表示项,$e.count$ 表示 x.FIT 中包含项 e 的事务的个数.

以表 1 中的数据集为例,说明 HYF 的构建和维护方法.表 1 的数据集有 5 个属性,分别为 A_1,A_2,A_3,A_4,C,其中,C 为类属性,第 1 列 T_{id} 表示每一个事务的序号.设用户给定的滑动窗口大小$|SW|=6$.

Table 1 Transaction dataset

表 **1** 事务数据集

T_{id}	A_1	A_2	A_3	A_4	C
T_1	a_1	b_1	d_1	e_2	c_2
T_2	a_1	b_1	d_1	e_1	c_2
T_3	a_2	b_1	d_1	e_2	c_1
T_4	a_3	b_2	d_1	e_2	c_1
T_5	a_3	b_3	d_2	e_2	c_1
T_6	a_3	b_3	d_2	e_1	c_2
T_7	a_2	b_3	d_2	e_1	c_1

(1) HYF 的构建方法

HYF 的构建过程就是算法插入事务的过程.当 FEP 算法读取了一个事务 $T=\{a_1,a_2,\ldots,a_n,c_i\}$时,按类值 c_i 将事务划分到类事务集合 C_i 中.然后,把事务 T 中除类值外的其他项作为一个新事务 T',即 $T'=\{a_1,a_2,\ldots,a_n\}$.接着,把新事务 T'分解为若干个子事务,$T_{a1}=\{a_1,a_2,\ldots,a_n\}$,$T_{a2}=\{a_2,\ldots,a_n\}$,$\ldots$,$T_{a(n-1)}=\{a_{n-1},a_n\}$,$T_{an}=\{a_n\}$,记为 $T_{ai}=\{a_i,\ldots,a_n\}, i\in\{1,2,\ldots,n\}$.完成映射后,FEP 算法将每个项 a_i 插入到 FI-list 中,并把该项对应的子事务 T_{ai} 插入到 a_i.FIT,然后更新 a_i.SFI_list.

以表 1 的数据集为例,FEP 算法读入第 1 个事务 $T_1=\{a_1,b_1,d_1,e_2,c_2\}$,首先把 T_1 划分到类事务集合 C_2 中.然后,将去掉类值 c_2 的新事务 $T_1'=\{a_1,b_1,d_1,e_2\}$映射为 4 个子事务 $T_{a1}=\{a_1,b_1,d_1,e_2\}$,$T_{b1}=\{b_1,d_1,e_2\}$,$T_{d1}=\{d_1,e_2\}$,$T_{e2}=\{e_2\}$,接着把 4 个项$\{a_1\},\{b_1\},\{d_1\},\{e_2\}$插入 FI-list,并把每个项对应的子事务插入到对应的 FIT 中,且更新相应的 SFI_list.图 1 是插入第 1 个事务后的子森林结构 HYF_2.

(2) HYF 的维护方法

HYF 的维护分为两个阶段:当窗口未满时,数据流 DS 有新事务到达,FEP 算法则直接把新事务插入到其类

值对应的子森林结构中,这也就是窗口的初始化阶段.

当 FEP 算法读入第 6 个事务后,滑动窗口已满.此时,类 c_1 的类事务集合 C_1 为{T_3,T_4,T_5},类 c_2 的类事务集合 C_2 为{T_1,T_2,T_6}.图 2 为此时的事务集合 C_2 所构成的子森林结构 HYF_2.

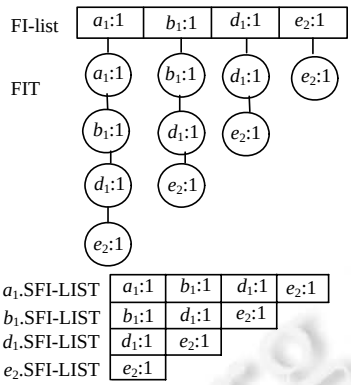


Fig.1 HYF_2 of C_2 after insert T_1

图 1 插入 T_1 后,C_2 构成的 HYF_2

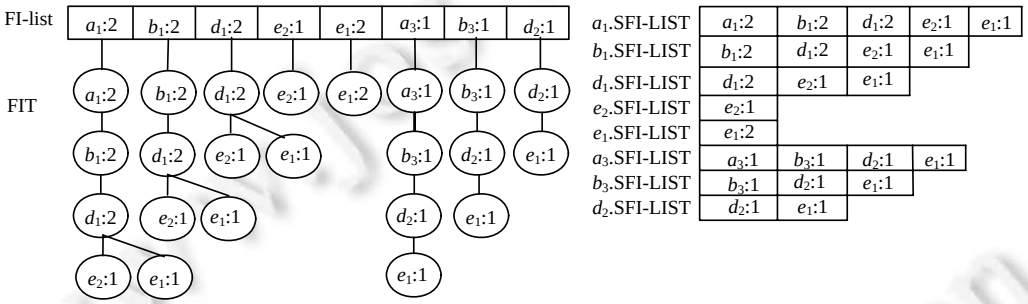


Fig.2 HYF_2 of C_2 after insert T_6

图 2 插入 T_6 后,C_2 构成的 HYF_2

当窗口已满时,FEP 算法要滑动窗口,也就是将窗口中最旧的数据删除,然后插入新读取的数据.删除旧数据 T_{old}={a_1,a_2,…,a_n,c_i}时,算法将维护对应的子森林结构 HYF_i.对 HYF_i 的维护,算法先将事务映射为 n 个子事务,然后把 FI-list 中各项 a_1,a_2,…,a_n 的计数减 1,并把各项对应的 a_i.FIT,i∈{1,2,…,n}中相关结点的计数减 1,且把 a_i.SFI_list 中相关元素的计数减 1.图 3 为删除 T_1 后 C_2 所构成的子森林结构 HYF_2.

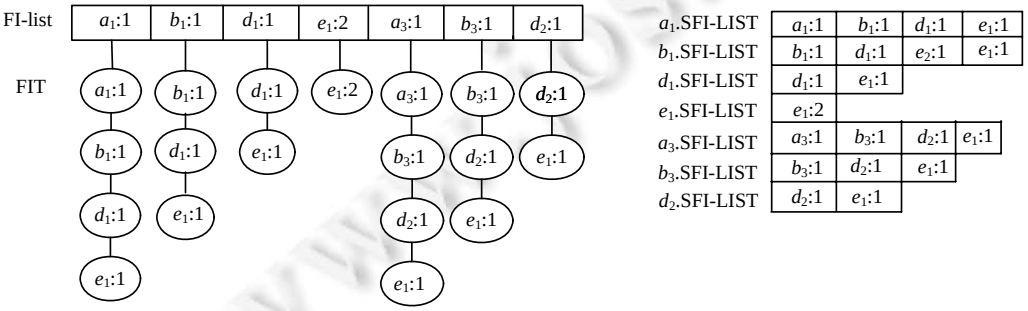


Fig.3 HYF_2 of C_2 after sliding the window

图 3 滑动窗口后 C_2 构成的 HYF_2

2.2 给定范围下数据流显露模式的挖掘

本文提出的基于显露模式的数据流分类算法EPDS采用半懒惰式的学习策略,通过两个步骤来挖掘显露模式:首先,有分类请求时,根据待分类事务 T_{test} 在混合森林结构 HYF 中分别抽取各类的频繁模式集合;然后,利用边界算法[3]挖掘每个类的显露模式.

2.2.1 给定范围下频繁模式的挖掘

有分类请求时,EPDS 算法根据待分类事务 T_{test} 在混合森林结构 $HYF=\{HYF_i\}$ 中分别抽取每个类的频繁模式集合.本文提出的抽取方法是一种给定抽取项范围的频繁项集抽取机制.

给定的数据流 DS,待分类事务 $T_{test}=\{x_1,x_2,\ldots,x_n\}$;当有分类请求时,在 HYF 的各个子结构 HYF_i 分别抽取与待分类事务具有相同项的项集集合.对于每一个子结构的项集抽取,算法分以下几步进行.

(1) 对于 $T_{test}=\{x_1,x_2,\ldots,x_n\}$ 中的每一项 x_i,其中,$i\in\{1,2,...,n\}$,在 FI_list 中找到具有相同项的结点;如果找到,则继续步骤(2);如果未找到,则换下一个项 x_{i+1} 在 FI_list 中继续查找,直至找到;若 T_{test} 中所有项在当前 HYF_i 的 FI_list 中都不存在具有相同项的元素,则返回空项集集合.

(2) 根据在 FI_list 中包含项 x_i 的元素找到项 x_i 对应的 x_i.SFI_list;对于 x_i 在事务 T_{test} 中后续项 $\{x_{i+1},x_{i+2},\ldots,x_n\}$,分别在 x_i.SFI_list 中找是否具有相同项的元素,若找到具有相同项的元素,则这些项连同 x_i 组成最长项集 $\{x_i,x_j,\ldots,x_k\}$,设项集 $\{x_j,\ldots,x_k\}$ 是 x_i.SFI_list 找到相同项的项集集合.

(3) 根据在 FI_list 中包含项 x_i 的元素找到项 x_i 对应的 x_i.FIT;检查找到的项集 $\{x_i,x_j,\ldots,x_k\}$ 在 x_i.FIT 中是否在同一分支上,设 $\{x_i,x_j,\ldots,x_k\}$ 在同一分支上,则项集 $\{x_i,x_j,\ldots,x_k\}$ 为找到的最佳项集.若 $\{x_i,x_j,\ldots,x_k\}$ 中存在不在同一条分支上的项,则对项集进行修剪,以删除不在同一分支上的项.

(4) 根据用户给定的支持度阈值 min_sup 对最佳项集 $\{x_i,x_j,\ldots,x_k\}$ 进行修剪,检查最佳项集中每一项的支持度是否超过 min_sup,对最佳项集进行修剪从而得到最佳频繁项集.

(5) 检查待分类事务,对未在最佳频繁项集中的项,重复执行步骤(1)~步骤(4),直到所有的项都被查找过.

2.2.2 显露模式挖掘

在抽取了每个类的频繁模式基础上,本文采用文献[3]提出的频繁模式集合的边界表示方法,并利用边界运算来挖掘显露模式.以类 c_i 为例,挖掘显露模式的方法分 3 步.

(1) 找到目标类和对立类数据集中频繁项集 FP_i 和 FP_i' 的最大频繁项集;

(2) 以空集为左边界和最大频繁项集为右边界的区间来表示频繁项集 FP_i 和 FP_i';

(3) 用边界运算来挖掘类 c_i 的显露模式 EP_i.

3 基于显露模式的数据流贝叶斯分类器

基于显露模式的贝叶斯分类器(EPDS)是半懒惰式分类器,在分类器的训练阶段对数据建立项集形式的密集表达;当有分类请求时,才对待分类事务建立特定的局部分类模型.

3.1 在数据流中估计联合概率

EPDS 使用从数据流中抽取的显露模式来建立贝叶斯分类器.对于贝叶斯联合概率的估计,算法依照条件独立模型来建立乘积近似值.

(1) 抽取到的项集决定乘积近似值的结构.

例如,给定的数据流 DS 包含属性 A_1,A_2,A_3,A_4,A_5 和类属性 C.c_i 是任意的类属性值,$T=\{a_1,a_2,\ldots,a_5\}$ 是待分类事务,为估计概率 $P(T,c_i)$ 的值,算法需要在混合森林结构 HYF_i 抽取项集.如果抽取项集集合为 $\{\{a_1,a_2\},\{a_3,a_4,a_5\}\}$,则建立的用于估计概率的乘积近似值为 $P(T,c_i)\approx P(c_i)P(a_1a_2|c_i)P(a_3a_4a_5|c_i)$.

(2) 对同一个待分类事务在各个类标值上分别抽取项集,即乘积近似值的结构与类标相关[21].

例如,设定类属性 C 有属性值 c_1,c_2.为预测待分类事务 T 的类标,EPDS 需要从子森林结构 HYF_1 和 HYF_2 中分别抽取项集集合来建立概率 $P(T,c_1),P(T,c_2)$ 的乘积近似值.

(3) 对于给定的类标值,乘积近似值的每一个乘积项中所隐含的属性集之间相互独立.

例如,联合概率 $P(T,c_i)$的乘积近似值为 $P(T,c_i)\approx P(c_i)P(a_1a_2|c_i)P(a_3a_4a_5|c_i)$,其中乘积项 $P(a_1a_2|c_i)$和 $P(a_3a_4a_5|c_i)$的属性集分别为$\{A_1,A_2\}$和$\{A_3,A_4,A_5\}$,则对于给定的类标 c_i,这两个属性集相互独立.

下面我们描述 EPDS 中抽取显露模式的原则.用于估计联合概率的项集必须满足的条件的优先级由高到低依次为:

(1) 选择的显露模式之间不包含重复项.两个显露模式不包含重复的项,即这两个显露模式的项的集合之间不存在交集.如果选择的显露模式之间存在重复的项,则构建的联合概率估计就不满足属性独立假设.

(2) 选择的显露模式增长率尽可能地大.由定义(9)可知,一个新事务 T 若包含从事务集合 C_i' 到 C_i 的支持度增长明显的显露模式,则事务 T 属于类 c_i 的概率高于非 c_i 类.

(3) 选择的显露模式尽可能长.显露模式的长度指显露模式中包含项的个数.抽取的显露模式越长,考虑的属性依赖关系就越多,就越能弱化朴素贝叶斯的强独立性假设,联合概率估计的乘积项也就越少.

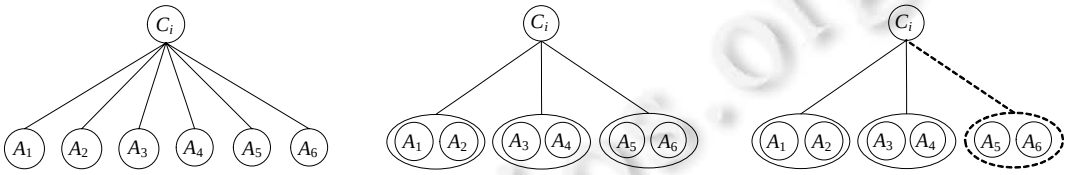
(4) 选择的显露模式的类支持度尽可能地大.在贝叶斯理论中,把新事务 T 预测为目标类 c_i,则要求联合概率 $P(T,c_i)$的值最大,即构成联合概率估计的乘积因子的值越大越好.因此,应该选择类支持度较大的显露模式来估计联合概率.用模式的支持度来估计联合概率时,支持度要用公式(9)做修正.

(5) 选择的显露模式集合尽可能地覆盖待分类事务所包含的项.数据流分类算法使用有限的内存来处理无限的数据,不能遍历整个数据集来抽取显露模式.另一方面,数据流还存在概念漂移的情况.尽管本文使用滑动窗口模型保证了分类模型在一定程度上避免概率漂移的发生,但可能存在抽取的显露模式的集合不能完全覆盖待分类事务的情况.在本文中,把未被覆盖到的属性作为一个整体(项集)来估计联合概率.这样能够最小化满足独立性假设的项集的数量,使得乘积近似更加符合条件独立模型.本文使用 Laplace 平滑[19]来处理未被覆盖的属性集合.

图 4(a)给出了测试事务 $T=\{a_1,a_2,a_3,a_4,a_5,a_6\}$在朴素贝叶斯的强条件独立假设模型下的情形.图 4(b)给出了测试事务 T 在基于显露模式弱条件独立假设模型下的理想情形.图 4(c)给出了测试事务 T 在基于显露模式弱条件独立假设模型下未被覆盖的情形.此时,属性 A_5 和 A_6 是未被覆盖的属性.本文中将未被覆盖的属性作为一个整体,用 Laplace 估计来处理,条件概率 $P(a_5a_6|c_i)$的值为

$$P(a_5a_6\mid c_i)=\frac{1}{count(c_i)+attnum(A_5)+attnum(A_6)} \tag{8}$$

其中,$attnum(A_5)$和 $attnum(A_6)$是属性 A_5 和 A_6 的取值个数,$count(c_i)$是窗口 SW 中类值为 c_i 的事务个数.



(a) 朴素贝叶斯强条件独立假设模型 (b) 基于显露模式理想的弱条件独立假设模型 (c) 基于显露模式未被覆盖的弱条件独立假设模型

Fig.4 Three cases of conditionally independent assumption model

图 4 条件独立假设模型的 3 种情形

3.2 显露模式支持度的调整

需要注意的是,本文优先选择增长率大的显露模式 X.尽管 X 从非 c_i 类的对立事务集合 C_i' 到 c_i 类的目标事务集合 C_i 的增长率大于阈值ρ,但是也存在 X 在类 c_i 的事务集合 C_i 中的类支持度 $Support_i(X)$却不高的情况.在数据流中,由于数据量是无限的,数据分布也是不断变化的,而我们只在当前窗口的事务中抽取显露模式,因此,

挖掘到具有低支持度高增长率度显露模式的情况更可能会出现.如果直接用 $Support_i(X)$来估计 $P(X)$,则会产生比较低的联合概率估计,造成分类结果不理想.

为了降低这种情况的影响,本文采取两个策略使得选择的显露模式更合适,使得估计的联合概率更合理.

(1) 选择的显露模式 X 在目标类 c_i 的事务集合 C_i 中要满足支持度阈值δ,保证显露模式 X 在目标类 c_i 的事务集合 C_i 中有一定的覆盖度.

(2) 用显露模式 X 的增长率 $Growth(X, C_i', C_i)$对在事务集合 C_i 中的支持度 $Support_i(X)$进行修正,用修正后的 $Suppor_{newi}(X)$来估计 $P(X)$.公式(9)给出了修正显露模式类支持度的方法.

$$Support_{newi}(X) = Support_i(X) \cdot Growth(X, C_i', C_i) \tag{9}$$

由于对支持度做了修订,因此最后需要对各类求得的概率做归一化处理.

3.3 分类器的训练

EPDS 分类器的训练阶段的主要工作是处理数据,即抽取并维护事务中的项,以方便后续分类模型的建立.所以在这一阶段的主要任务是建立混合森林结构,当有新的数据生成时,更新滑动窗口和相应的混合森林结构.算法 1 描述了在训练阶段使用滑动窗口模型处理数据的完整过程.

算法 **1**. FEP: Find Emerging Pattern Algorithm on Data Stream.

输入:数据流 $DS=[T_1,T_2,\ldots,T_n,\ldots)$,滑动窗口大小$|SW|$.

输出:混合森林结构 $HYF=\{HYF_i\}$,其中,i 为对应的类标值.

```
01: for all T in DS
02:     //w 是当前窗口中事务的个数.
03:     If w<|SW|
04:         w=w+1;
05:         HYFBuilding(T,HYF_i);
06:     end if
07:     Else
08:         w=w−1;
09:         deletefromHYF(T_old,HYF_i);
10:         w=w+1;
11:         HYFBuilding(T,HYF_i);
12:     end if
13: end for
```

当数据流中的新事务到来时,算法先判断窗口是否已满:若窗口未满,则将事务加入窗口,并根据事务的类标将事务加入相应的子森林结构(第 4 行、第 5 行);若窗口已满,则在窗口和相应的子森林结构中删除最旧事务的信息,再将新事务加入窗口和对应的子森林结构中(第 8 行~第 11 行).

3.4 类标预测

EPDS 是基于显露模式的数据流贝叶斯分类器,分类器采用半懒惰式的学习策略.对于一个待分类事务 T_{test},EPDS 在每个类值所对应的子森林结构 HYF_i 中抽取频繁模式,然后利用边界运算方法挖掘每个类的显露模式集合,最后从显露模式集合中选取模式来构建局部分类器.算法 2 描述了 EPDS 预测类标的过程.

算法 **2**. $ClassPrediction(HYF,T_{test},min\_sup_i,min\_sup_j)$.

输入:$HYF=\{HYF_i\}$,测试事务 T_{test},类 c_i 的最小用户支持度阈值 min_sup_i,非类 c_i 的最小用户支持度阈值 min_sup_j.

输出:待分类事务 T_{test} 的类标 c.

```
01: for all c_i in C
```

02: $finalItemset=\varnothing$;
03: while $T_{test}\neq$null do
04: //分别从 HYF_i 和其他非 c_i 的子森林结构 HYF_j 中抽频繁模式
05: $FpatternC_i=selectFPItemset(T_{test},HYF_i,min\_sup_i)$;
06: $FpatternC_j=selectFPItemset(T_{test},HYF_j,min\_sup_j)$;
07: //挖掘显露模式
08: $EPsets=MBD\_LLBORDER(FpatternC_j,FpatternC_i)$;
09: //从显露模式集合中选择估计联合概率的显露模式
10: $bestFpattern=selectEPset(EPsets)$;
11: //移除测试事务项集集合中已被选为最佳项集的项
12: $T_{test}=T_{test}/bestFpattern$;
13: $finalItemset=finalItemset\cup bestFpattern$;
14: end while
15: //l 是 $finalItemset$ 中的子集
16: $P(T_{test},c_i)=P(c_i)\Pi l\in finalItemsetP(l|c_i)$;
17: end for
18: return the class c_i with maximal $P(T_{test},c_i)$;

算法首先对待分类事务在每个类标下挖掘频繁模式集合,然后利用边界运算挖掘每个类的显露模式集合,再从每个类的显露模式集合中分别抽取用于估计联合概率的显露模式集合(第 3 行~第 14 行).并根据抽取到的显露模式集合分别估计每个类的联合概率值(第 16 行).再比较各个联合概率值的大小,从而对待分类事务的类标进行预测(第 18 行).

EPDS 分类器的存储空间依赖于滑动窗口中项的数量.假设当前窗口中项的数量为 m,则算法的空间复杂度为 $O(m^2)$.EPDS 分类器的运算主要是项的查询和比较.随着窗口中项的数量的增加,算法的运行时间也会相应地增加.同时,EPDS 是半懒惰式分类器,会有比懒惰式分类器快的处理速度.

4 实验分析

本文进行的实验主要从分类精度对算法的性能进行评价.本文采用 UCI 机器学习库中的真实数据集以及由数据生成器生成的合成数据集来进行实验.实验平台是 Massive Online Analysis(MOA)[22].实验在 3.60GHz、Intel(R) Core(TM)i7-4790 CPU、8G 内存、Windows 7 系统的计算机上进行.

4.1 数据集

本文使用的 3 个真实数据集分别是来自 UCI 数据集(http://archive.ics.uci.edu/ml/)的 EEG,Firm 和 MAGIC;5 个合成数据集分别是由数据生成器 Agrawal Generator[21]生成的 AGRAWAL、Random RBF Generator 生成的 RBF、random RBF drift 生成的 RBF Drift、SEA Generator[23]生成的 SEA 和 STAGGER Generator[14]生成的 STAGGER.每个合成数据集都由 100 000 条事务组成.数据生成器都来自 MOA 平台.各数据集的主要特征在表 2 中给出.

表 2 中的属性 Attribute 不包含类属性.从表中可以看出,EEG,MAGIC,AGRAWAL,RBF,RBF Drift 和 SEA 中包含的有连续型属性.而本文的 EPDS 分类器要求数据均为名称型属性,在算法中对相关数据集进行了离散化预处理操作.本文的离散化操作采用的是 Weka(http://www.cs.waikato.ac.nz/ml/weka/)平台上的最小化信息熵的启发式算法[24].

表 2 也给出了各数据集离散化后的特征信息.表中的 Items 也不包含“类属性-值”对.另外需要注意的是,在离散化后的数据集中,存在一些属性在离散化后只有 1 个值的问题,这些属性会干扰贝叶斯理论的概率估计.对于这种情况,我们做了第 2 步预处理,即删除掉这样的属性.AGRAWAL 和 SEA 数据集离散后的属性个数分别变

为 4 和 2.

Table 2 Real and synthetic data sets

表 2 真实数据集和合成数据集

Dataset	Transaction	Attribute			Items	Classes
		Continuous	Nominal	Discretized		
EEG	14 980	14	0	14	82	2
Firm	10 800	0	19	19	38	2
MAGIC	19 020	10	0	10	79	2
AGRAWAL	100 000	5	3	4	49	2
RBF	100 000	10	0	10	254	2
RBF Drift	100 000	10	0	10	254	2
SEA	100 000	3	0	2	23	2
STAGGER	100 000	0	3	3	9	2

4.2 实验模型

本文对算法性能的评价采用的是预测误差估计法(prequential error estimators)[25].本文对所有数据集,将滑动窗口的窗口大小 w 固定为 w=10%×n,其中,n 是数据集的事务个数;将模式在目标类和对立类数据集合中的最小支持度阈值固定为同一个较小的值,min_sup_i=min_sup_j=0.01%×w,其中,w 是当前窗口大小.对于显露模式的支持度从对立类数据集合到目标类数据集合的增长率,本文在选择用来构建贝叶斯概率估计的显露模式时,每次都选择增长率尽可能大的模式,因而没有再设置最小增长率阈值.

4.3 结果分析

为了对比 EPDS 分类器的性能,本文选择 MOA 平台上的多种类型的分类器以提高实验的全面性.实验所用的对比分类器包括:数据流贝叶斯分类器 Naïve Bayes(NB),Naïve Bayes Multinomial(NBM)[26];懒惰式分类器 k-NN[27],k-NN with PAW(PAW)[27];关联型分类器 Rule Classifier(RC)[28],Rule Classifier Naïve Bayes(RCNB)[28]以及其他的数据流分类器 Hoeffding Tree(HT)[29],Hoeffding Option Tree(HOT)[30],SGD,ORTO[31],FIMTDD[32].

表 3 展示的是各分类器在实验数据集上的分类精度.其中,第 2 列~第 12 列分别给出了各对比分类器的分类精度,第 13 列给出了 EPDS 的分类精度.

Table 3 Accuracy comparison with other classifiers

表 3 与其他分类器的分类精度对比

Datasets	NB	NBM	k-NN	PAW	RC	RCNB	HT	HOT	SGD	ORTO	FIMT-DD	EPDS
EEG	83.60	82.30	**91.20**	91.10	90.60	90.40	79.60	86.30	90.70	90.70	90.70	87.10
Firm	99.70	99.80	**100.0**	**100.0**	**100.0**	**100.0**	**100.00**	**100.0**	**100.0**	**100.0**	**100.00**	**100.0**
MAGIC	47.50	68.50	**100.0**	**100.0**	**100.0**	**100.0**	**100.00**	**100.0**	**100.0**	–	–	**100.0**
AGRAWAL	94.9	67.38	90.38	90.3	91.86	93.92	94.94	94.95	67.42	67.44	67.44	**95.79**
RBF	75.77	68.77	79.14	**81.07**	53.36	72.89	78.72	78.87	68.48	49.91	49.91	78.80
RBF Drift	75.77	68.77	79.14	**81.07**	53.36	72.89	78.72	78.87	68.48	49.91	49.91	78.80
SEA	86.46	64.14	78.76	80.87	70.06	83.88	87.83	87.95	66.65	35.83	35.83	**89.60**
STAGGER	**100.0**	92.70	**100.0**	**100.0**	**100.0**	**100.0**	**100.00**	**100.0**	88.50	88.50	88.50	**100.0**
Average	82.96	76.55	89.83	90.55	82.41	89.25	89.98	90.87	81.28	68.90	68.90	**91.26**

从表 3 可以看出,EPDS 的平均分类精度优于其他各种分类器.下面我们具体分析与每种分类器的对比情况.

(1) 与数据流贝叶斯分类器相比,EPDS 分类器的分类精度只在 STAGGER 数据集上与 NB 算法相持平,而整体上都优于 NB 和 NBM.

(2) 与懒惰式的数据流分类器相比,k-NN 和 PAW 算法只分别在 EEG,RBF 和 RBF Drift 数据集上有更优的精度,而 EPDS 分类器在大多数数据集上的分类精度都持平或优于 k-NN 和 PAW 算法,分类精度的均值均优于 k-NN 和 PAW 算法.

(3) 与关联型数据流分类器相比,EPDS 分类器在大多数数据集上的分类精度和所有数据集上的分类精度均值都优于 RC 和 RCNB 算法.

(4) 与其他数据流分类器进行比较,EPDS 分类器在大多数数据集上的分类精度和所有数据集上的分类精度均值也都优于 HT,HOT,SGD,ORTO 和 FIMTDD 算法.

从表 3 中还可以看到,ORTO 和 FIMT-DD 算法在 3 个合成数据集 RBF,RBF Drift 和 SEA 上的分类精度相对于其他分类器有明显降低.ORTO 和 FIMT-DD 算法是基于回归树的算法,算法中采用基于标准差的分割度量标准.而实验中对 3 个合成数据集的数值型属性进行离散化的过程中会有一定程度的数据信息损失,使得 ORTO 和 FIMT-DD 算法在这 3 个数据集上的分类精度降低.同时我们也注意到,在真实数据集 MAGIC 上,NB 算法的分类精度也有明显的降低.这是因为 MAGIC 数据集中属性之间有较大的关联度,与 NB 算法中属性间的强独立性假设不一致,而 EPDS 算法则减弱了 NB 的强独立性假设,获得了较高的分类精度.

分析认为,EPDS 算法能够有较好的性能是因为存在以下 3 个方面的优势.

(1) 采用半懒惰式学习方式,针对待分类事务建立局部的分类模型,使模型更具有针对性,有助于分类器性能的提升.

(2) 使用具有很好分类性能的显露模式基于条件独立模型来估计联合概率的近似值,同时考量了模式在不同类之间的支持度变化,并对用于估计联合概率近似值的支持度进行了修正,降低了增长率较高而支持度较低的显露模式对分类性能的影响.

(3) 对同一待分类事务,在不同的类标值上分别建立联合概率乘积近似结构,提高了算法对不同类的数据的适应性.

以上 3 个优势使得分类模型建立的过程能够更好地获取数据中属性之间的依赖关系,以及更多地考虑模式在类与类之间的区分度,进而提高分类正确率.

5 结 论

本文提出了一个基于显露模式的数据流贝叶斯分类器 EPDS.EPDS 采用滑动窗口机制,提出混合森林结构 HYF 对数据进行密集存储.EPDS 采用半懒惰式学习策略,根据待分类事务来选取每个类下的显露模式集合,并建立局部的分类模型.本文设计了属性独立假设下的显露模式抽取机制,使得在估计联合概率时既考虑了属性之间的关联关系,也考虑了模式在目标类和对立类数据集合间的区分度.实验结果表明,EPDS 比其分类器有较好的分类精度.

本文介绍的模型主要关注于二值分类问题,并且也仅关注于和各类数据流分类算法在分类精确度上的比较.将来的工作方向包括进行分类模型自身相关参数对分类性能影响的研究,如窗口大小、最小支持度,并可以把该模型扩展到多类值数据流分类问题上,以及延伸到数据流概念漂移问题上等.

References:

[1] Keogh EJ, Pazzani MJ. Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches. In: Proc. of the 7th Int'l Workshop on Artificial Intelligence and Statistics. Fort Lauderdale: Society for Artificial Intelligence and Statistics, 1999. 225−230.

[2] Aggarwal CC. Data Streams: Models and Algorithms. Berlin: Springer-Verlag, 2007.

[3] Dong G, Li J. Efficient mining of emerging patterns: Discovering trends and differences. In: Proc. of the 5th Int'l Conf. on Knowledge Discovery and Data Mining (KDD'99). San Diego: ACM Press, 1999. 43−52. [doi: 10.1145/312129.312191]

[4] Chen CC, Shi HX, Fan M. Aggregating EP-based classifiers to classify data streams. Journal of Computer Research and Development, 2006,43(z3):376−381 (in Chinese with English abstract).

[5] Sammut C, Webb GI. Encyclopedia of Machine Learning and Data Mining. Berlin: Springer-Verlag, 2017. 389−392. [doi: 10.1007/978-1-4899-7502-7]

[6] Zhang P, Gao BJ, Zhu XQ, Guo L. Enabling fast lazy learning for data streams. In: Proc. of 2011 IEEE the 11th Int'l Conf. on Data Mining (ICDM 2011). Vancouver: IEEE Computer Society, 2011. 933−941. [doi: 10.1109/ICDM.2011.63]

[7] Li HF, Shan MK, Lee SY. DSM-FI: An efficient algorithm for mining frequent itemsets in data streams. Knowledge and Information Systems, 2008,17(1):79−97. [doi: 10.1007/s10115-007-0112-4]

[8] Lewis DD. Naive (Bayes) at forty: The independence assumption in information retrieval. In: Proc. of the 10th European Conf. on Machine Learning (ECML-98). Berlin, Heidelberg: Springer-Verlag, 1998. 4−15. [doi: 10.1007/BFb0026666]

[9] Friedman N, Goldszmidt M. Building classifiers using Bayesian networks. In: Proc. of the 13th National Conf. on Artificial Intelligence (IAAI'96). Portland: MIT Press, 1996. 1277−1284.

[10] Baralis E, Cagliero L, Garza P. Enbay: A novel pattern-based Bayesian classifier. IEEE Trans. on Knowledge and Data Engineering, 2013,25(12):2780−2795. [doi: 10.1109/TKDE.2012.197]

[11] Baralis E, Cagliero L. RIB: A robust itemset-based Bayesian approach to classification. Knowledge-Based Systems, 2014,71:366−375. [doi: 10.1016/j.knosys.2014.08.015]

[12] Fan H, Ramamohanarao K. A Bayesian approach to use emerging patterns for classification. In: Proc. of the 14th Australasian Database Conf. (ADC 2003). Adelaide: Australian Computer Society, 2003. 39−48.

[13] Meretakis D, Wijthrich B. Extending naive Bayes classifiers using long item sets. In: Proc. of the 5th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD'99). San Francisco: ACM Press, 1999. 165−174. [doi: 10.1145/312129.312222]

[14] Schlimmer JC, Granger RH. Incremental learning from noisy data. Machine Learning, 1986,1(3):317−354. [doi: 10.1023/A:1022810614389]

[15] Li GH, Chen H. Mining the frequent patterns in an arbitrary sliding window over online data streams. Ruan Jian Xue Bao/Journal of Software, 2008,19(10):2585−2596 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/2585.htm [doi: 10.3724/SP.J.1001.2008.02585]

[16] Alavi F, Hashemi S. DFP-SEPSF: A dynamic frequent pattern tree to mine strong emerging patterns in streamwise features. Engineering Applications of Artificial Intelligence, 2015,37:54−70. [doi: 10.1016/j.engappai.2014.08.010]

[17] García-Borroto M, Martínez-Trinidad JF, Carrasco-Ochoa JA. A survey of emerging patterns for supervised classification. Artificial Intelligence Review, 2014,42(4):705−721. [doi: 10.1007/s10462-012-9355-x]

[18] Liu Y, Li JZ, Zhu JH. A novel graph classification approach based on frequent closed emerging patterns. Journal of Computer Research and Development, 2007,44(7):1169−1176 (in Chinese with English abstract). [doi: 10.1360/crad20070711]

[19] Domingos P, Pazzani M. On the optimality of the simple Bayesian classifier under zero-one loss. Machine Learning, 1997,29(2):103−130. [doi: 10.1023/A:1007413511361]

[20] Wang Z, Fan H, Ramamohanarao K. Exploiting maximal emerging patterns for classification. In: Proc. of the 17th Australian Joint Conf. on Artificial Intelligence (AUS-AI 2004). Cairns: Springer-Verlag, 2004. 1062−1068. [doi: 10.1007/978-3-540-30549-1_102]

[21] Agrawal R, Imielinski T, Swami A. Database mining: A performance perspective. IEEE Trans. on Knowledge and Data Engineering, 1993,5(6):914−925. [doi: 10.1109/69.250074]

[22] Bifet A, Holmes G, Kirkby R, Pfahringer B. MOA: Massive online analysis. Journal of Machine Learning Research, 2010,11:1601−1604.

[23] Street WN, Kim YS. A streaming ensemble algorithm (SEA) for large-scale classification. In: Proc. of the 7th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD 2001). San Francisco: ACM Press, 2001. 377−382. [doi: 10.1145/502512.502568]

[24] Fayyad UM, Irani KB. Multi-Interval discretization of continuous-valued attributes for classification learning. In: Proc. of the 13th Int'l Joint Conf. on Artificial Intelligence (IJCAI'93). Chambéry: Morgan Kaufmann Publishers, 1993. 1022−1029.

[25] Gama J, Sebastiao R, Rodrigues PP. Issues in evaluation of stream learning algorithms. In: Proc. of the 15th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD 2009). Paris: ACM Press, 2009. 329−338. [doi: 10.1145/1557019.1557060]

[26] McCallum A, Nigam K. A comparison of event models for naive Bayes text classification. In: Proc. of the AAAI-98 Workshop on Learning for Text Categorization, Vol.752. Madison: AAAI Press, 1998. 41−48.

[27] Bifet A, Read J, Pfahringer B, Holmes G. Efficient data stream classification via probabilistic adaptive windows. In: Proc. of the 28th Annual ACM Symp. on Applied Computing (SAC 2013). Coimbra: ACM Press, 2013. 801−806. [doi: 10.1145/2480362.2480516]

[28] Gama J, Kosina P. Learning decision rules from data streams. In: Proc. of the 22nd Int'l Joint Conf. on Artificial Intelligence (IJCAI 2011). Barcelona: AAAI Press, 2011. 1255−1260.

[29] Hulten G, Spencer L, Domingos P. Mining time-changing data streams. In: Proc. of the 7th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD 2001). San Francisco: ACM Press, 2001. 97−106. [doi: 10.1145/502512.502529]

[30] Pfahringer B, Holmes G, Kirkby R. New options for hoeffding trees. In: Proc. of the 20th Australasian Joint Conf. on Artificial Intelligence (AUS-AI 2007). Gold Coast: Springer-Verlag, 2007. 90−99. [doi: 10.1007/978-3-540-76928-6_11]

[31] Zliobaite I, Bifet A, Pfahringer B, Holmes G. Active learning with evolving streaming data. In: Proc. of the European Conf. on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2011). Athens: Springer-Verlag, 2011. 597−612. [doi: 10.1007/978-3-642-23808-6_39]

[32] Ikonomovska E, Gama J, Džeroski S. Learning model trees from evolving data streams. Data Mining and Knowledge Discovery, 2011,23(1):128−168. [doi: 10.1007/s10618-010-0201-y]

附中文参考文献:

[4] 陈崇超,施鸿喜,范明.集成基于 EP 的分类器用于分类数据流.计算机研究与发展,2006,43(z3):376−381.

[15] 李国徽,陈辉.挖掘数据流任意滑动时间窗口内频繁模式.软件学报,2008,19(10):2585−2596. http://www.jos.org.cn/1000-9825/19/2585.htm [doi: 10.3724/SP.J.1001.2008.02585]

[18] 刘勇,李建中,朱敬华.一种新的基于频繁闭显露模式的图分类方法.计算机研究与发展,2007,44(7):1169−1176. [doi: 10.1360/crad20070711]

杜超(1983－),男,河南三门峡人,博士生,讲师,主要研究领域为数据挖掘,机器学习.

江晶晶(1992－),女,硕士,主要研究领域为数据挖掘,机器学习.

王志海(1963－),男,博士,教授,博士生导师,CCF 专业会员,主要研究领域为数据挖掘,机器学习.

孙艳歌(1982－),女,博士生,讲师,CCF 专业会员,主要研究领域为数据挖掘,机器学习.