

The Study and Implementation of Discretization Algorithm Based on Information Entropy

Chengxia Liu^{1,2}, Minling Zhu²

¹Beijing Key Laboratory of Internet Culture and Digital Dissemination Research, Beijing

²Computer School, Beijing Information and Technology University, Beijing

Email: cecilia7812@163.com

Received: 26th, 2019; accepted: Dec. 12th, 2019; published: Dec. 19th, 2019

Abstract

The values range of continuous attributes is divided by discretization algorithm into several parts, each of which corresponds to its own discrete symbol. The reasonable discretization determines the accuracy of information expressing. This article studies and implements a discretization algorithm based on information entropy. The set S is divided by measuring the importance of the breakpoint by giving the breakpoint information entropy. First, a set of candidate breakpoint attributes of continuous attribute are calculated. Secondly, a breakpoint from the set of candidate breakpoints is selected to add the breakpoint with the smallest value of information entropy to the set of breakpoints, which breaks up the set S into two parts. The third determines the breakpoint for each set of instances until the partitioning for set S satisfies the minimum discrimination length criterion. In the last part of the article, the correctness and validity of the algorithm are verified by experiments, and test as well as comparison of different groups of data is given.

Keywords

Information Entropy, Discretization, Breakpoint

基于信息熵的离散化算法的研究与实现

刘城霞^{1,2}, 朱敏玲²

¹北京信息科技大学网络文化与数字传播北京市重点实验室, 北京

²北京信息科技大学计算机学院单位, 北京

Email: cecilia7812@163.com

收稿日期: 2019年11月26日; 录用日期: 2019年12月12日; 发布日期: 2019年12月19日

摘要

离散化算法将连续属性的取值范围划分为很多个小的区间,每个区间都对应着自己的离散化符号,合理的离散化能够更准确的表达信息。本课题研究并实现了一种基于信息熵的离散化算法,通过赋予断点信息熵来度量断点的重要性从而对集合S进行划分。首先计算连续的属性的候选断点属性集,其次从候选断点集合中选取一个使信息熵最小的断点加入到断点集合中,该断点把集合S分成了两个部分,之后对于每一个子集合确定断点直到对于集合S的划分足够表达不同信息,满足最小区分长度准则完成。本文最后用实验验证了此算法的正确性和有效性,并对多组数据进行了测试和比较。

关键词

信息熵, 离散化, 断点

Copyright © 2019 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

离散化是数据挖掘中非常重要的一步,它属于数据预处理阶段,是对连续属性数据进行的分析处理。具体离散化算法会把连续属性的数据取值范围划分为很多个小区间,划分的原则分为有监督和无监督,有监督的分为等宽度,等频率[1]等,而无监督就是指没有指定要求的情况下的根据具体算法具体分析。而基于信息熵的离散化算法即是其中一种无监督的离散化方法,它通过判断断点信息熵的大小来划分实例集,达到离散化的目的。

2. 信息熵基本概念

信息是个很抽象的概念,起初人们对信息的多少无法说清楚,直到香农在1948年信息论中提出的信息熵[2]这一概念才使的量化度量信息的问题得以解决。在信息论中,信息熵被视为接收到的每条信息种包含的信息量的平均值,也被称为信源熵、平均自信息量。

用 X 表示随机变量,随机变量的取值为 (x_1, x_2, \dots, x_n) , $p(x_i)$ 表示事件 x_i 发生的概率,且有 $\sum p(x_i) = 1$ 。定义事件 x_i 的信息量是它发生概率对数的负数,记为 $I(x_i)$,即: $I(x_i) = -\log(p(x_i))$ 。

而 $H(X)$ 为随机变量 X 的平均信息量,即 X 的信息熵。

$$H(X) = E[-\log P(x_i)] = -\sum_{i=1}^N P(x_i) \log P(x_i)$$

其中 $P(x_i)$ 表示事件 x_i 发生的先验概率,且 $\sum P(x_i) = 1$ 。

另外,公式中的 \log 的底和信息熵的单位有关。被广泛应用的是以2为底,它的单位是bit。当使用以e为底,单位为Nat。

3. 最小信息熵的离散化方法

在智能化信息处理中,经常遇到的技术难点主要是离散化问题和数据的不完整性问题。在应用粗糙集理论来处理信息决策表时,需要满足的是决策表中的值应该是离散的。不管是连续属性还是离散化的

数据, 都是需要进行离散化, 连续属性在处理前必须进行离散化处理; 对于离散化的数据, 可以使用离散化方法将本来也就是离散的数据合并或者抽象得到更好的离散值。离散化算法应该满足:

- 1) 离散化后的信息系统的空间维数应该尽量少, 即离散化后信息系统剩余的属性个数尽量少。
- 2) 离散化后的信息丢失尽量少。

1993年, Fayyad 和 Irani 于提出最小信息熵离散化方法[3]。离散的门槛边界值由候选区间的类信息熵来选择。子集 S_1 和子集 S_2 的信息熵为 $Ent(S_1)$ 和 $Ent(S_2)$, 则由断点 T 关于属性 A 的类信息熵可以表示为

$$E(A, T; S) = |S_1|/|S| \cdot Ent(S_1) + |S_2|/|S| \cdot Ent(S_2)$$

$|S|$ 表示集合中元素个数。

对于给定的属性 A , 将使得 $E(A, T; S)$ 值最小的点作为离散化的划分点, 并将其记为 T_{min} 。划分点 T_{min} 将区间划分为两个部分, 样本集合也被分为两个集合 S_1 和 S_2 。在分别计算 S_1 和 S_2 的所有划分点的时候, 假设 T_1 和 T_2 分别为 S_1 和 S_2 的最好的划分点, 而它们所对应的类信息熵分别为 $E(A, T_1, S_1)$ 和 $E(A, T_2, S_2)$; 如果 $E(A, T_1, S_1) > E(A, T_2, S_2)$, 就继续对 S_1 进行划分, 反之则是 S_2 。也就是说, 哪个大就划分哪个。然后递归调用他们, 直到满足递归停止条件。停止条件是利用最小区分长度准则作为判断递归离散化是否停止, 就是说集合 S 中的递归的划分区间, 当满足下列条件时停止,

$$Gain(A, T; S) < \log_2(N-1)/N + \Delta(A, T; S)/N$$

其中: N 是集合 S 中的样本数量, $Gain(A, T; S) = Ent(S) - Ent(A, T, S)$,

$$\Delta(A, T; S) = \log_2(3^k - 2) - [k \cdot Ent(S) - k_1 \cdot Ent(S_2) - k_2 \cdot Ent(S_2)]$$

其中 k_1 和 k_2 分别是集合 S_1 和 S_2 中的类别数量, 即 $k = |S|$, $k_1 = |S_1|$, $k_2 = |S_2|$ 。

4. 基于最小信息熵的离散化算法的设计及实现

基于信息熵的离散化算法有很多, 比如文献[4][5]等。本文采取的基于最小信息熵的离散化算法的主要步骤为: 第一, 计算一个连续属性的候选断点属性集。第二, 从第一步得到的候选断点属性集中选取一个使信息熵的值最小的点, 此断点把实例集合划分为两个部分, 将此断点加入到断点集合中。第三, 确定每个实例集合的断点, 划分实例集合 S 直到它满足递归停止条件则退出程序, 完成离散化工作。该算法是一个比较基础的算法, 它每次仅从单个条件属性的候选断点中选择断点[6], 得到的最终结果断点数目较多, 但时间复杂度较低。但本文中是以针对该算法采用多组数据进行比较, 实验结果表明此算法是有效的。

具体算法描述表示如下:

步骤 1: 计算连续属性的候选断点属性集。

步骤 2: 从第一步得到的候选断点属性集中选取一个使信息熵的值最小的点, 此断点把实例集合划分为两个部分, 将此断点加入到断点集合中。信息熵计算方法如下: 对于一个给定的实例的集合 S , 一个属性 A , 一个分类的边界 T (把 S 分为 S_1 和 S_2 两个部分), 那么由分类边界 T 确定的分类的信息熵为:

$$E(A, T; S) = |S_1|/|S| \cdot Ent(S_1) + |S_2|/|S| \cdot Ent(S_2)$$

信息熵计算公式为 $Ent(S) = -\sum_{j=1}^k P_j \log(P_j)$

步骤 3: 对每个实例集合都调用步骤 2 中的方法进行划分, 确定断点。递归停止条件为:

$$Gain(A, T; S) < \log_2(N-1)/N + \Delta(A, T; S)/N$$

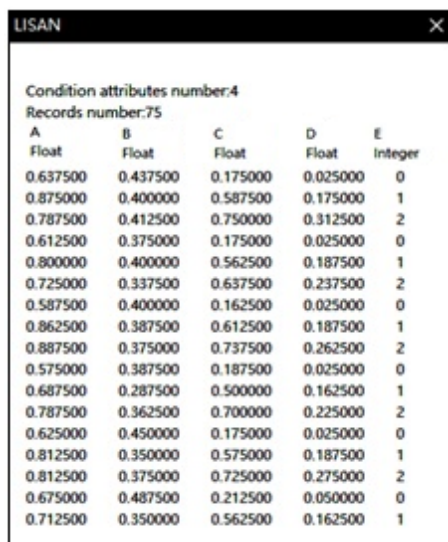
具体算法伪代码如下:

```
void GetBreakPointInAttr(CArray<float,float>&breakpoint_array, CArray<STAT,STAT>&stainfo,
                        int low,int high)
{ if(low>=high)
  { return; }
  float d_entropy, d_temp; //存储信息熵值
  int i_index=low,i_total; //i_index 记录最小熵断点
  i_total=GetTotal(stainfo,low,high); //i_total 表示样本值个数
  d_entropy=GetEntropy(stainfo,low,low,high,i_total); //计算信息以 low 位置为断点时的信息熵值
  for (int i=low+1;i<high;i++)
  { d_temp=GetEntropy(stainfo,i,low,high,i_total); //获得以 i 位置为断点时的信息熵值
    if(d_temp<d_entropy) //比较信息熵, 取最小值
    { i_index=i; d_entropy=d_temp; }
  }
  float d_val=float((stainfo.GetAt(i_index).d_value+stainfo.GetAt(i_index+1).d_value)/2); //得到断点值
  AddToArrayByOrder(d_val,breakpoint_array); //按照从小到大的顺序插入断点队列
  if(CanStop(stainfo,i_index,low,high)) //判断是否达到递归的停止条件
  { return; }
  GetBreakPointInAttr(breakpoint_array,stainfo,low,i_index); //对左边的划分递归
  GetBreakPoint_InNonStrAttr(breakpoint_array,stainfo,i_index+1,high); //对右边的划分递归
}
```

5. 结果分析

5.1. 简单例子

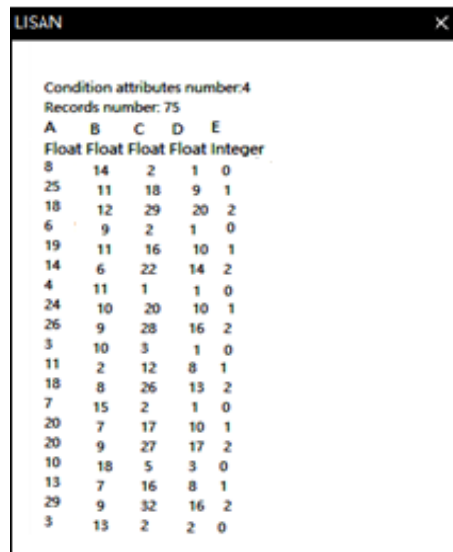
测试一个较为简单的例子, 条件属性: 4 个; 决策属性: 1 个; 记录条数: 75 条; 属性名称: A, B, C, D, E; 属性类型: float, float, float, float, integer。离散化前后结果如图 1~2 所示:



LISAN				
Condition attributes number:4				
Records number:75				
A	B	C	D	E
Float	Float	Float	Float	Integer
0.637500	0.437500	0.175000	0.025000	0
0.875000	0.400000	0.587500	0.175000	1
0.787500	0.412500	0.750000	0.312500	2
0.612500	0.375000	0.175000	0.025000	0
0.800000	0.400000	0.562500	0.187500	1
0.725000	0.337500	0.637500	0.237500	2
0.587500	0.400000	0.162500	0.025000	0
0.862500	0.387500	0.612500	0.187500	1
0.887500	0.375000	0.737500	0.262500	2
0.575000	0.387500	0.187500	0.025000	0
0.687500	0.287500	0.500000	0.162500	1
0.787500	0.362500	0.700000	0.225000	2
0.625000	0.450000	0.175000	0.025000	0
0.812500	0.350000	0.575000	0.187500	1
0.812500	0.375000	0.725000	0.275000	2
0.675000	0.487500	0.212500	0.050000	0
0.712500	0.350000	0.562500	0.162500	1

Figure 1. Before discretization

图 1. 离散化前



Condition attributes number:4					
Records number: 75					
A	B	C	D	E	
Float	Float	Float	Float	Integer	
8	14	2	1	0	
25	11	18	9	1	
18	12	29	20	2	
6	9	2	1	0	
19	11	16	10	1	
14	6	22	14	2	
4	11	1	1	0	
24	10	20	10	1	
26	9	28	16	2	
3	10	3	1	0	
11	2	12	8	1	
18	8	26	13	2	
7	15	2	1	0	
20	7	17	10	1	
20	9	27	17	2	
10	18	5	3	0	
13	7	16	8	1	
29	9	32	16	2	
3	13	2	2	0	

Figure 2. After discretization

图 2. 离散化后

测试结果表明, 对于 float 类型的属性进行离散化后得到了离散化后的属性符号, 并且该例中本是离散属性的 E 不需要再离散化。由于数据量小, 离散化时间几乎为 0。

5.2. 性能分析

本次测试主要测试了 75~4000 条数据, 具体运行时间折线图如图 3 所示。

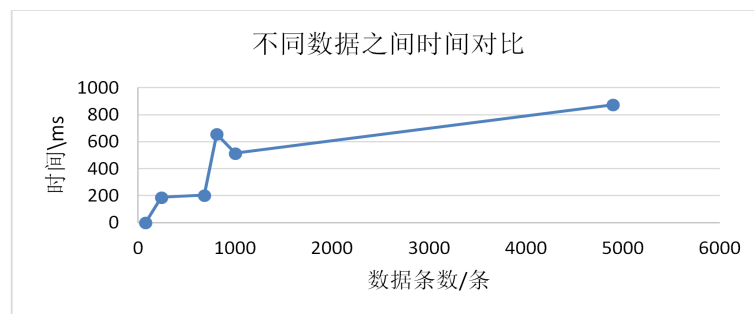


Figure 3. The comparison of time with different data scale

图 3. 运行时间比较图

从图上可以看出, 随着数据量的增大, 离散化时间也在增大。但增长又不是线性的, 这是因为离散化时间不但和记录的条数有关, 还和断点的个数、需要离散化属性的个数以及属性的类型等有关。

6. 总结与展望

本文研究并实现了基于最小信息熵的离散化算法, 它可以从这些含有连续属性的数据集中取得好的数据样本, 得到简洁且有效的规则, 如此可以方便数据在后续挖掘中的处理, 并帮助企业及用户更有效的挖掘需要的数据。

基金项目

本项目得到网络文化与数字传播北京市重点实验室开放课题资助; 促进高校内涵发展 - 科研水平提

高项目(5221823410)资助。

参考文献

- [1] 侯利娟, 王国胤, 聂能, 等. 粗糙集理论中的离散化问题[J]. 计算机科学, 2000, 27(12): 89-94.
- [2] Shannon, C.E. (1948) A mathematical theory of communication. *The Bell System Technical Journal*, **27**, 379-423. <https://doi.org/10.1002/j.1538-7305.1948.tb00917.x>
- [3] Fayyad, U.M. and Irani, K.B. (1993) Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93)*, Chambéry, 28 August-3 September 1993, 1022-1027.
- [4] 谢宏, 程浩忠, 牛东晓. 基于信息熵的粗糙集连续属性离散化算法[J]. 计算机学报, 2005, 28(9): 1570-1574.
- [5] 高建国, 崔业勤. 基于信息熵理论连续属性离散化方法[J]. 微电子学与计算机, 2011, 28(7): 187-189.
- [6] 刘业政, 焦宁, 姜元春. 连续属性离散化算法比较研究[J]. 计算机应用研究, 2007, 24(9): 28-30+33.