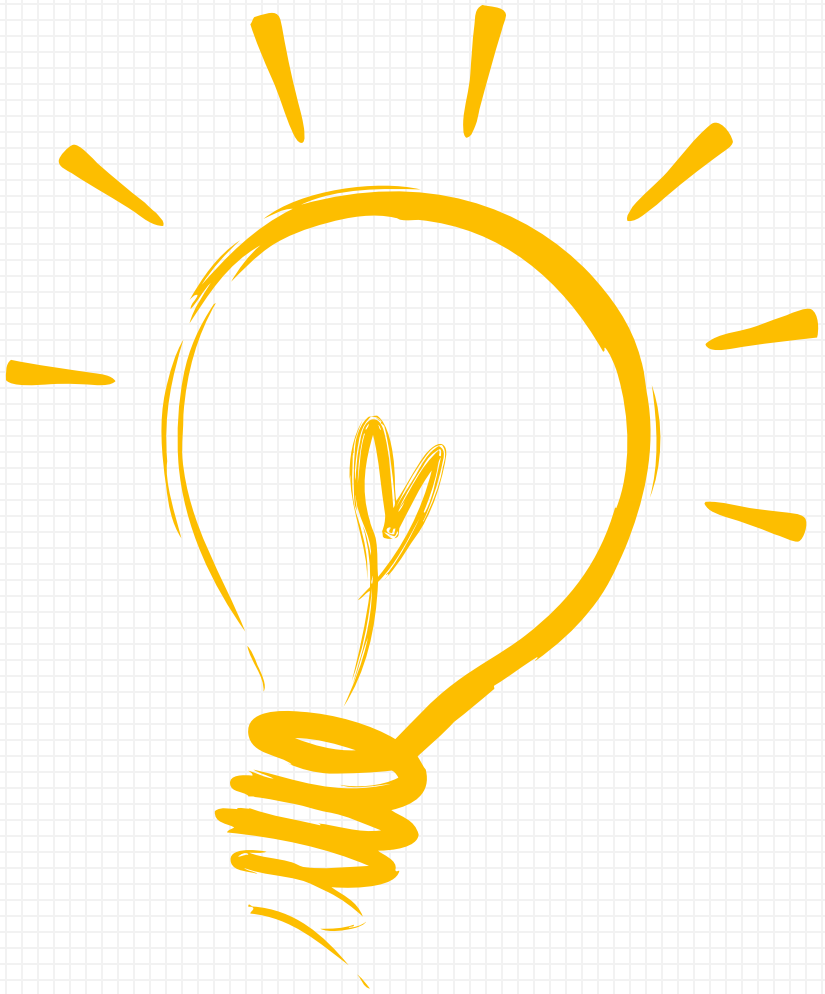

Thinking
Space
Good Idea
idea
question
E=mc²
Engineering
Invent
business growth
Solution
Learning
Study
ecology
Innovation
ideas
Renewable energy
Education
research
Science
Chemical

数据挖掘算法

关联规则

CONTENTS



di yi zhang jie

第一章节

算法背景

1.1 啤酒和尿布的故事[1]

沃尔玛的 “啤酒与尿布”是营销界经典的案例，该案例是正式刊登在1998年的《哈佛商业评论》上面的，这应该算是目前发现的最权威报道。

20世纪90年代的美国沃尔玛超市中，沃尔玛的超市管理人员分析销售数据时发现了一个令人难于理解的现象：在某些特定的情况下，“啤酒”与“尿布”两件看上去毫无关系的商品会经常出现在同一个购物篮中，这种独特的销售现象引起了管理人员的注意，经过后续调查发现，这种现象出现在年轻的父亲身上。

在美国有婴儿的家庭中，一般是母亲在家中照看婴儿，年轻的父亲前去超市购买尿布。父亲在购买尿布的同时，往往会顺便为自己购买啤酒，这样就会出现啤酒与尿布这两件看上去不相干的商品经常会出现在同一个购物篮的现象。

1.2 故事引发的思考[1]

“啤酒与尿布”是通过人工的观察分析得到的一种关系。面对如此多的超市消费记录，我们怎么才能高效的发现物品之间的相关性?

换句话说，我们能不能在数学上寻求这么一种算法，能在计算机上运行，从而高效的去找到这种规则?

1993年美国学者Agrawal 提出通过分析购物篮中的商品集合，从而找出商品之间关联关系的关联算法，并根据商品之间的关系，找出客户的购买行为。艾格拉沃从数学及计算机算法角度提出了商品关联关系的计算方法——Apriori算法。

di yi zhang jie

第二章节

算法原理

2.1 案例引导

考虑如下一家杂货店简单交易清单：

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

共有7条交易记录。A B C 等代表的是不同商品。

2.2 基本概念

Def 1 事务集：如右表，{ABCD,ABCF,BDEF….}，所有的流水记录构成的集合

Def 2 记录（事务）：ABCD 叫做一条记录（事务）

Def 3 项目（项）：A , B , C ….叫做一个项目（项）

Def 4 项目集（项集）：由项组成的集合，如{A,B,E,F}，{A,B,C}就是一个项集

Def 5 K项集：项集中元素的个数为K，如{A,B,E,F}就是4项集

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.2 基本概念

Def 6 支持度（Support）：

$$Sup(X)=\frac{\text{某个项集 X 在事务集中出现的次数}}{\text{事务集中记录的总个数}}$$，简单理解就是概率（频率）

如 X = {A,C} 则 $Sup\ (X) = \frac{4}{7} = 0.57$

Def 7 置信度（Confidence）：

$$Con\ (X=>Y) = \frac{Sup\ (XY)}{Sup\ (x)}$$，简单理解就是条件概率

条件概率:$P(Y|X) = \frac{p\ (XY)}{p\ (x)}$

如 X = {A}， Y = {C} 则 $Con\ (X=>Y) = \frac{Sup\ (XY)}{Sup\ (x)} = \frac{4/7}{5/7} = \frac{4}{5} = 0.8$

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.2 基本概念

Def 8 最小支持度（min_support）：人为规定的一个支持度。

Def 9 最小置信度（min_confidence）：人为规定的一个置信度。

Def 10 提升度： Lift（A->B） = $\frac{\text{con}\ (\text{A}->\text{B})}{\text{sup}\ (B)}$,理解为B在A发生的基础上再发生的概率与B单独发生概率的比值。

Def 11 频繁K项（目）集：满足最小支持度的K项集。

Def 12 候选K项（目）集：用来生成频繁K项集的K项集。（不等价与所有K项集）

最小支持度为0.5 $\text{X} = \{\text{A,C}\}$ 则 $\text{Sup}\ (\text{X}) = \frac{4}{7} = 0.57 \geq 0.5$ ， 称$\{A, C\}$是一个频繁2项集

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.3 两个定理

Theorem 1 : 如果X是一个频繁K项集，则它的所有子集一定也是频繁的。

Theorem 2 :如果X不是K-1项频繁，则它一定不是频繁K项集。

理解：

比方我们在某个公司内部，想升职，必须满足在该等级绩效前三名，然后必须是一级一级升。

假如有5个级别，职员，组长，车间主管，部门主管，总经理。

现在有一个人，他是车间主管，那你就会知道，他一定是当过职员，当过组长，而且还满足当时是绩效前三名。

同理如果你有没有当过车间主管，或者即使当了车间主管，但不是绩效前三名，那你肯定当不上部门主管。

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.4 算法流程

Step1：令K = 1，计算单个商品（项目）的支持度，并筛选出频繁1项集（最小支持度为0.3 = 2.1/7 ）

Step2：（从K=2开始）根据K-1项的频繁项目集生成候选K项目集，并进行预剪枝

Step3：由候选K项目集生成频繁K项集（筛选出满足最小支持度的k项集）

重复步骤2和3，直到无法筛选出满足最小支持度的集合。（第一阶段结束）

Step4：将获得的最终的频繁K项集，依次取出。同时计算该次取出的这个K项集的**所有真子集**，然后以排列组合的方式形成关联规则，并计算规则的置信度以及提升度，将符合要求的关联规则生成提出。（算法结束）

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.4 算法流程

Step1：令K = 1，计算单个商品（项目）的支持度，并筛选出频繁1项集（最小支持度为0.3 = 2.1/7 ）

{A},{B},{C},{D},{E},{F} 的支持度分别是：0.71，0.86，0.71，0.57，0.57，0.29 。

所以得到的频繁1项集为{A},{B},{C},{D},{E}

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.4 算法流程

Step2：（K=2）根据K-1项的频繁项目集生成候选K项目集，并进行预剪枝

频繁1项集为{A},{B},{C},{D},{E}

如何根据K-1项的频繁项目集生成候选K项目？（最小支持度为0.3 = 2.1/7 ）

用K-1项集排列组合：

{A,B} {A,C} ~~{A,D}~~ ~~{A,E}~~ {B,C} {B,D} {B,E} {C,D} ~~{C,E}~~ ~~{D,E}~~

4/7 4/7 2/7 2/7 4/7 3/7 4/7 3/7 2/7 2/7

（这里暂时没有用到预剪枝）

Step3：由候选K项目集生成频繁K项集（筛选出满足最小支持度的k项集）

{A,B} {A,C} {B,C} {B,D} {B,E} {C,D} 得到频繁2项集

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.4 算法流程

Step2：（K=3）根据K-1项的频繁项目集生成候选K项目集，并进行预剪枝

{A,B} {A,C} {B,C} {B,D} {B,E} {C,D}（最小支持度为0.3 ）

用K-1项集排列组合：

{A,B,C} {A,B,D} {A,B,E} ~~{A,B,C,D}~~ ~~{A,B,C,E}~~ {A,C,D} {B,C,D} {B,C,E} {B,D,E} ~~{B,E,C,D}~~

组合K项候选集的时候注意，两个集合要有K-2项相同

预剪枝：因为{A,D} {A,E} {C,E} {D,E} 都不是频繁2项集，所以在候选3项集中，含有这些2项集的3项集，一定不是频繁3项集。因此：经过剪枝的候选3项集为

{A,B,C} ~~{A,B,D}~~ ~~{A,B,E}~~ ~~{A,B,C,D}~~ ~~{A,B,C,E}~~ ~~{A,C,D}~~ {B,C,D} ~~{B,C,E}~~ ~~{B,D,E}~~ ~~{B,E,C,D}~~

即： {A,B,C} {B,C,D} 是最终的候选3项目集。

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.4 算法流程

Step3：由候选K项目集生成频繁K项集（筛选出满足最小支持度的k项集）

Sup（ABC） = 3/7　Sup（BCD） = 2/7

频繁3项集为{A,B,C}。另外由于无法再生成候选4项目集，算法第一阶段结束。

Step4：将获得的最终的频繁K项集，依次取出。同时计算该次取出的这个K项集的**所有真子集**，然后以排列组合的方式形成关联规则，并计算规则的置信度以及提升度，将符合要求的关联规则生成提出。（算法结束）

获得的最终的频繁K项集：{A,B,C}….（这里只有一个，实际上可能有很多）

依次取出：第一个是{A,B,C}，获取其所有真子集

{A} {B} {C} {B,C} {A,C} {A,B} {B,C}

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.4 算法流程

以排列组合的方式形成关联规则

{A} ->{B}、{A} ->{C}、{B} ->{C}、 {B} ->{A}、{C} ->{A}、{C} ->{B}

{A} ->{B,C}、 {B} ->{A,C} 、{C} ->{A,B} 、{A,B} ->{C}、 {A,C} ->{B}、 {B,C} ->{A}

思考：{A,C} 和{A,B}为什么不组合?

计算置信度及提升度，以{A} ->{B,C}为例

$$\text{Con}\ (\ \{A\} \text{->} \{B,C\}\ ) = \frac{Sup(\{A,B,C\})}{Sup(\{A\})} = \frac{3}{5} = 0.6$$

$$\text{Lift}\ (\ \{A\} \text{->} \{B,C\}\ ) = \frac{\text{Con}\ (\ \{A\} \text{->} \{B,C\}\ )}{Sup(\{B,C\})} = \frac{0.6}{4/7} = 1.05$$

因此 {A} ->{B,C} 就是一条比较令人信服的关联规则。

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

2.4 算法流程

计算置信度及提升度，以{B,C} ->{A} 为例

$$\text{Con}\ (\ \{B,C\} \rightarrow \{A\}\ ) = \frac{Sup(\{A,B,C\})}{Sup(\{B,C\})} = \frac{3}{4} = 0.75$$

$$\text{Lift}\ (\ \{B,C\} \rightarrow \{A\}\ ) = \frac{\text{Con}\ (\{B,C\} \rightarrow \{A\})}{Sup(\{A\})} = \frac{0.75}{5/7} = 1.05$$

注意到 {A} -> {B,C} 的置信度是0.6，所以两条关联规则还是有区别的

同理，计算其他关联规则的置信度和提升度即可。最后输出满足条件的规则。

单号	商品
1	A B C D
2	A B C E
3	B D E F
4	B C D E
5	A C D F
6	A B C
7	A B E

di san zhang jie

第三章节

算法分析

3.1 算法分析[2]

优点： Apriori算法采用逐层搜索的迭代方法，算法简单明了，没有复杂的理论推导，也易于实现。

缺点：

1.时间复杂度大，注意到，每次计算某个K项集的支持度时，总是要扫描一遍事务集才可以，而现实中，事务集中记录的个数往往很大，因此算法运行效率效率不高。

2.算法可能产生大量的候选项集。（排列组合）

3.在频繁项目集长度变大的情况下,运算时间显著增加。

4.采用唯一支持度,没有考虑各个属性重要程度的不同等。

di si zhang jie

第四章节

算法拓展

4.1 算法拓展

针对经典Apriori算法的缺点，很多前辈基于经典算法提出或者改进出了更加高效的，更加合理全面，适用面更加广的关联规则挖掘算法，比如FP-growth算法。

以下是几篇这方面的论文，大家感兴趣可以看看。

[1]冯玉才,冯剑琳.关联规则的增量式更新算法[J].软件学报,1998(04):62-67.

[2]龙舜,蔡跳,林佳雄.一个基于演化关联规则挖掘的个性化推荐模型[J].暨南大学学报(自然科学与医学版),2012,33(03):264-267.

[3]张勇,李树青,程永上.基于频次有效长度的加权关联规则挖掘算法研究[J].数据分析与知识发现,2019,3(07):85-93.

di wu zhang jie

第五章节

案例实操

5.1 案例实操

这一部分在下次视频中讲解，我们使用Python中基于经典Apriori的第三方库，来操作一下。

需要做的准备是：

首先要有Python环境，最好可以安装上pycharm。

然后安装一下必要的库（以后也需要的）：如numpy，pandas，xlrd，Apriori库，scikit-learn库等

如果你现在没有Python环境以及pycharm，或者不清楚第三方库如何安装，以及pip文件如何设置，pycharm的一些设置等，可以看一下我之前的基于Python实现网络爬虫的视频的第一课：

https://www.bilibili.com/video/BV1WV411U7LQ

同时如果有一些其他问题，可以联系我

QQ： 1366420642，Q群： 1019030249

欢迎大佬萌新加入

参　考　资　料

【1】https://cloud.tencent.com/developer/article/1105331

【2】https://blog.csdn.net/icebns/article/details/105535366

感谢上述链接文章作者。

THANKS

谢谢观看