MapReduce 框架下的朴素贝叶斯算法并行化研究①

幸莉仙, 黄慧连

(华北电力大学大学 经济管理系, 保定 071003)

摘 要: 研究朴素贝叶斯算法 MapReduce 的并行实现方法, 针对传统单点串行算法在面对大规模数据或者参与分类的属性较多时效率低甚至无力承载大规模运算, 以及难以满足人们处理海量数据的需求等问题, 本文在朴素贝叶斯基本理论和 MapReduce 框架的基础上, 提出了一种基于 MapReduce 的高效、廉价的并行化方法. 通过实验表明这种方法在面对大规模数据时能有效提高算法的效率, 满足人们处理海量数据的需求.

关键词: 朴素贝叶斯; MapReduce; 并行化; 云计算

Parallelization of Naive Bayes Algorithm Under MapReduce Framwork

XING Li-Xian, HUANG Hui-Lian

(School of Business and Administration, North China Electric Power University, Baoding 071003, China)

Abstract: This article focused on the realization of the parallelization of Naive Bayes. When it comes to large-scal data or multi-attributes, the traditional singal node algorithm has a low efficiency,or even is unable to host large-scale computing. All of these make the traditional algorithm cannot fit the need to deal with massive data. Therefore, based on the basic theory of Naive Bayes and the framework of MapReduce, this paper proposed a parallelization method of Naive Bayes, which is efficient and cheap.At the end, it is proved by experiments that this method can effectively improve the efficiency of the algorithm so as to meet the need of peoople to deal with massive data.

Key words: Naive Bayes; MapReduce; parallelization; cloud computing

贝叶斯分类是一种基于统计学的分类方法, 它主要利用概率统计知识进行分类, 一般用于解决 “在给定训练实例集的情况下, 判定新实例的类别”这类问题[1]. 贝叶斯提供了一种能自然表示因果信息的方法用来发现数据之间的潜在关系, 尽管关联规则也是一种挖掘数据之间潜在关系的方法, 但是关联规则不能进行逆向的推理, 而贝叶斯网络支持某种情况下原因到结果, 结果到原因的正向推理和逆向判断. 贝叶斯网络算法的核心是遍历数据计算各个数据项的先验概率. 现阶段贝叶斯网络算法主要包括朴素贝叶斯、TNA 贝叶斯和无约束贝叶斯, 其中朴素贝叶斯算法虽看似简单, 当随着日益膨胀的数据和分类属性的增加, 尤其随着信息系统的普及, 各种数据以每年 80%的速度增长, 因此当面临 TB 级甚至更大的数据量时, 传统的单点运行的方法效率会非常低, 其资源消耗也会使得现阶段的计算机难以承载, 甚至不能提供有时间限制的服务.

随着计算机技术的飞速发展, 云计算已成为分布式计算未来发展的方向, 由 Google 提出的 MapReduce 编程框架是云计算中的代表性技术, 它适用于分布式处理大规模数据集, 计算效率极高[2].

针对传统算法的无法处理或者无法高效处理海量数据的缺陷, 本文通过研究 MapReduce 基本框架和朴素贝叶斯算法, 在 Apache 提供 Hadoop 分布式文件系统(HDFS)和 MapRedce 并行计算框架的基础上, 以 Hbase 作为数据分布式存储的数据库, 研究朴素贝叶斯的并行化算法实现. 通过实验表明这种并行算法与传统单点串行算法相比, 这种实现方式能处理传统算法不能处理的海量数据, 并具有较高的效率.

① 收稿时间:2012-07-10;收到修改稿时间:2012-08-22

1 朴素贝叶斯算法

贝叶斯是一种解决不确定性知识推理的方法，根据对变量的要求条件不同贝叶斯一般分为有约束贝叶斯和无约束贝叶斯，朴素贝叶斯属于典型的有约束贝叶斯网络，它假设各变量之间是相互独立的关系，即给定类变量的状态，每个属性变量与其他属性变量是独立的. 尽管实际生活中这种假设很难成立，但很多研究和实践表明：即使假设前提不能满足，但是在很多领域朴素贝叶斯与决策树、K-Means 等经典算法相比仍具有其优越性.

朴素贝叶斯算法的基本描述如下:

设有数据集 T，共分成 N 类 L_1，L_2，...,L_N，对于数据集中的每个样本 X 有 n 个属性，则 X 可以表示为 $X=\{x_1,x_2,x_3...x_n\}$；$x_1,x_2,x_3...x_n$ 分别表示 n 个属性 A_1,A_2，A_3，...A_n 的取值.

(1)对于一个未知分类的样本 X，朴素贝叶斯算法将把 X 归为条件 X 下具有最高后验概率的类. 也就是说当且仅当：$P(C_i \mid X) > P(C_j \mid X), 1 \le j < N$，且$i \ne j$ 时，X 归为 C_i 类.

(2)由贝叶斯定理我们知道:

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)}$$

在上式中，对于所有类别 C,P(X)为常数，因此只需 $P(X \mid C_i)P(C_i)$ 最大.

(3)在 $P(X \mid C_i)P(C_i)$ 中:

$$P(C_i) = \frac{\text{数据集中}C_i\text{类的样本数}}{\text{数据集的总样本数}}$$

当给定了数据集数量或者分类属性的数量较多时，对于 P(X|Ci)计算的开销可能非常大，但是朴素贝叶斯假设个属性之间是相互独立的，因此有:

$$P(X \mid C_i) = P(x_1 \mid C_i)P(x_2 \mid C_i)...P(x_k \mid C_i)$$
$$= \prod_{j=1}^{n} P(x_j \mid C_i)$$

因此对于任意未知分类的样本 X，当且仅当下面条件是我们将 X 归为 C_i 类:

$$P(X \mid C_i)P(C_i) > P(X \mid C_p)P(C_p)$$
$$\text{其中} 1 \le i, p \le N, \text{且} i \ne p$$

朴素贝叶斯算法的计算主要集中在样本训练阶段，它需要计算出每个属性的每个取值的先验概率. 虽然计算简单，但是当面对大数据量或者参与分类的属性增加时，需要消耗的计算机资源也会成倍的增加. 因此尽管在处理小规模的数据时，传统的算法简单易行，但是随着训练样本或者分类属性的增加，传统的单点串行算法在处理海量数据时还有相当的局限性，这是本文的出发点和需要改进的地方.

2 MapReduce编程框架

云计算采用类似 MapReduce 的编程模式，MapReduce 是一种处理海量数据的并行编程模型和计算框架，用于大规模数据的并行计算. 它采用一种“分而治之”的思想，把大的任务分解成若干个较小的任务分发给各个节点来执行，再将各个节点的执行结果进行汇总，从而得到最终的结果[3-6]. 这种处理过程分为 Map 过程和 Reduce 两个阶段.

在 Map(映射)阶段，MapReduce 框架将任务的输入数据先分割成若干固定大小的块(Split)，对于 Split 又分解成一批键值对(Key1, Value1)传给 Map 函数；每个节点的 Map 函数对于每组键值对进行处理后，形成新的键值对(Key2, Value2),并按照 Key2 值相同的进行汇总，形成(Key2, list(Value2))，传给 Reduce 作为 Reduce 的出入. 一般来说，Map 会将 Key2 值相同的键值对传给相同的节点来进行 Reduce 阶段的处理.

在 Reduce 阶段，Map 输出的(Key2, list(Value2))成为 Reduce 阶段的输入，对于输入作相应处理后会得到键值对(Key3, Value3)，根据用户需要输出到 HDFS 或者 HBase 数据库等指定的位置.

为使得 MapReduce 的数据处理流程更加形象，下图 1 描绘了 MapReduce 模型的计算流程.

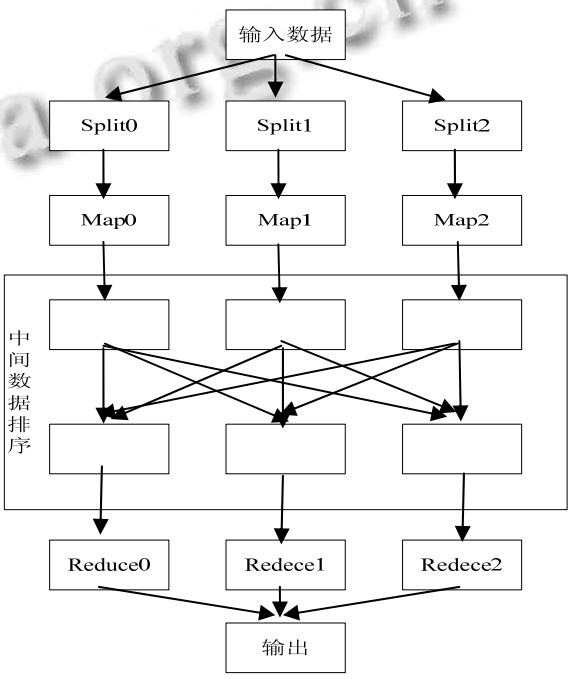

图 1 MapReduce 数据处理流程

3 朴素贝叶斯的MapReduce实现

个贝叶斯算法的实现包括训练数据和测试数据两基本的阶段, 数据项的统计主要在训练数据生成模型阶段, 而且通常测试数据只是少数, 因此本文主要介绍朴素贝叶斯算法中生成贝叶斯网络模型的 MapReduce 实现方法. 下面以不同人群是否买电脑为例来具体介绍朴素贝叶斯训练结算的并行算法. 下表 1 为 HBase 存储的的训练数据, 以此为输入数据来训练贝叶斯模型.

表 1　训练数据表

RowKey	Income	Credit_rating	Age	Student	Buy_computer
1	Hige	Fair	31-40	No	Yes
2	Low	Fair	<=30	Yes	No
3	Medium	Excellent	<=30	Yes	Yes
4	Medium	Excellent	>40	No	No
5	High	Fair	31-40	Yes	Yes

Income,Credit_rating,Age,Student 分别为属性值, Buy_computer 表示类别. 对于每个属性我们都构建一个向量, 该向量的长度为其对应属性不同取值的个数; 例如对于 Income 我们构建一个长度为三的向量 Vincome[3]=[high,medium,low]; 若 Vincome=[1,0,0]则表示记录 Income 属性取值为 High.

3.1 Map 阶段

输入: 存储在 HBase 中的数据以键值对的形式传给 Map 函数, 其中 Key0 为每条记录的 RowKey, 记录的内容为 Value0.

处理: 将 Value0 中属性字段的取值转换成向量, 以第一条记录为例:

V1=[{1},{1 0 0},{1 0},{1 0 0},{0 1}]

其中 V1[0]={1}表示记录出现的次数, 以便 Reduce 时统计各类的总数; V1[1]={1 0 0}表示该记录 Income 属性为 High, 也表示 High 属性出现的次数为 1; V1[2]、V1[3]、V1[4]以此类推.

输出: Map 输出依然是键值对的形式, 不过此时 Key1 为每条记录的类别标号; Value1 为 V1 转换成字符串后的形式

3.2 Reduce 阶段

输入: Map 的输出即为 Reduce 的输出, 此时 Key1, Value1 即为 Reduce 的输入

处理: 对于输入的 Value1 转换成二位数组 meta[n+1][]; n 表示属性的个数. 仍然以第一条数为例, 将 V1 转换成:

$$\text{meta}[][]=\begin{bmatrix}1 & & \\ 1 & 0 & 0\\ 1 & 0 & \\ 1 & 0 & 0\\ 0 & 1 & \end{bmatrix}$$

然后对各个 meta 按照类别进行对应位置的加和, 得到 summeta[][].

输出: Reduce 阶段输出每个类别汇总后的 summeta[][]及其类别标号.

3.3 分类阶段

这样通过 MapReduce 的处理最终输出两个数组分别为:

$$\text{sum}meta\_yes[][]=\begin{bmatrix}3 & & \\ 2 & 1 & 0\\ 1 & 2 & \\ 1 & 2 & 0\\ 2 & 1 & \end{bmatrix}$$

$$\text{summeta}\_no[][]=\begin{bmatrix}2 & & \\ 0 & 1 & 1\\ 1 & 1 & \\ 1 & 0 & 1\\ 1 & 1 & \end{bmatrix}$$

其中 summeta_yes[0][0]和 summets_no[0][0]分别表示训练样本中属于 yes 和 no 的样本数, 则

P(X=Yes)=summeta_yes[0][0]/(summeta_yes[0][0]+Summeta_no[0][0])=3/(2+3)=0.6;

P(X=No)=summeta_no[0][0]/(summeta_yes[0][0]+summeta_no[0][0])=2/(2+3)=0.4;

数组其余行代表该类别下的属性, 每行中的数代表属性各种取值在该类别出现的次数, 将 summeta_yes 和 summets_no 各行归一化变得等到个属性取值的先验概率. 如下所示:

$$\text{sum}meta\_yes[][]=\begin{bmatrix}1 & & \\ 2/3 & 1/3 & 0\\ 1/3 & 2/3 & \\ 1/3 & 2/3 & 0\\ 2/3 & 1/3 & \end{bmatrix}$$

$$\text{summeta}\_no[][]=\begin{bmatrix} 1 & & \\ 0 & 1/2 & 1/2 \\ 1/2 & 1/2 & \\ 1/2 & 0 & 1/2 \\ 1/2 & 1/2 & \end{bmatrix}$$

下面预测记录 R1: Income= “High”, Credit_rating= “Excellent”,Age= “<=30”, Student= “No”, 则

P(Buy_computer="Yes"|X)

=P(X|Buy_computer="Yes")*P(X=Yes)

=2/3*1/3*1/3*1/3*0.6=8/135

P(Buy_computer="No"|X)

=P(X|Buy_computer="No")*P(X=No)

=0*1/2*1/3*1/3*0.4=0

P(Buy_computer="Yes"|X)>P(Buy_computer="No"|X); 因此我们预测 R1 会买电脑.

4 实验过程及分析

为验证并行算法的效率, 搭建若干个节点的集群环境, 每个节点的配置都是 CPU 为 Intel Pentium P6200; 主频为 2.40GHz; 内存为 4G; 且每个节点都是百兆以太网, Hadoop 版本 0.20.2、java 版本 1.6 ; HBase 版本 0.20.3, 布置好程序运行环境.

第一组实验: 选取一组数据, 约 245M, 包含 100 万条左右的数据, 选取 10 个属性进行实验. 分别将 MapReduce 中参与计算的节点数分别设置 2, 4, 6, 8, 算法运行的时间依次为 321s, 206s, 158s, 131s 如图 2 所示.

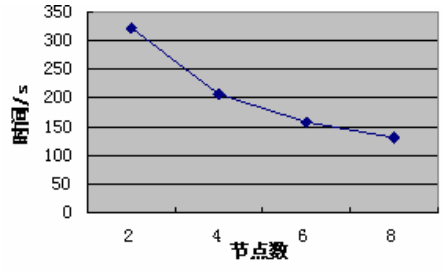

图 2 不同节点数的运行时间

从上图容易看到, 随着节点数的增加, 处理相同的数据运行时间在减少, 因此当面对更大的数据量时, 我们就可以通过增加节点来提高算法的执行效率.

第二组实验: 将传统的朴素贝叶斯算法和并行化的贝叶斯算法进行测试, 比较结果如下表 2 所示:

表 2 串行算法与并行算法性能比较

实现次数	数据大小(MB)	记录条数	串行算法运行时间(S)	并行算法运行时间(S)
1	0.6	480	12	46
2	2.3	1520	68	57
3	16.4	12630	654	93
4	64	54902	内存不足	126
5	256	245034	内存不足	185

从上表可以看出, 在处理的数据量足够小时, 单点串行算法的效率要比并行算法更高, 这是由于 MapReduce 本身 job 调度以及数据的 Split 和重组等需要耗费一定的时间. 但是随着数据量的不断增加, 并行算法的优越性也逐渐显现出来, 不仅很大程度提高了效率, 而且能完成单点串行算法不能承载的数据量的运算, 为处理海量数据提供参考.

5 结语

本文通过对朴素贝叶斯算法和云计算 MapReduce 框架进行分析, 详细介绍了 MapReduce 编程框架下的朴素贝叶斯的并行算法, 并通过实验表明该并行的朴素贝叶斯算法实现简单, 当面对海量数据时其效率远远比传统的串行算法高, 是一种解决海量数据分析处理的有效方法.

参考文献

1 廖芹,郝志峰,陈志宏.数据挖掘与数学建模.北京:国防工业出版社,2010.292–300.

2 陈康,郑纬民.系统实例与研究现状.软件学报,2009,(5):1337–1348.

3 刘鹏,黄宜华,陈卫卫.实战 Hadoop—开启通向云计算的捷径.北京:电子工业出版社,2011.

4 余楚礼,肖迎元,尹波.一种基于 Hadoop 的并行关联规则算法.天津理工大学学报,2011,27(1):25–32.

5 张圣.一种基于云计算的关联规则 Apriori 算法.通信技术,2011,44(6):141–142.

6 Isard M, Budiu M, Yu Y, et al. Dryad:Distributed data-parallel programs from sequential building blocks.Proc.of the 2nd European Conf.on Computer Systems(EuroSys),2007,59–72.