Rrr

**随机算法课程**

**实验报告**

**实验一：**用minHash进行集合相似性连接

姓名：王艺达

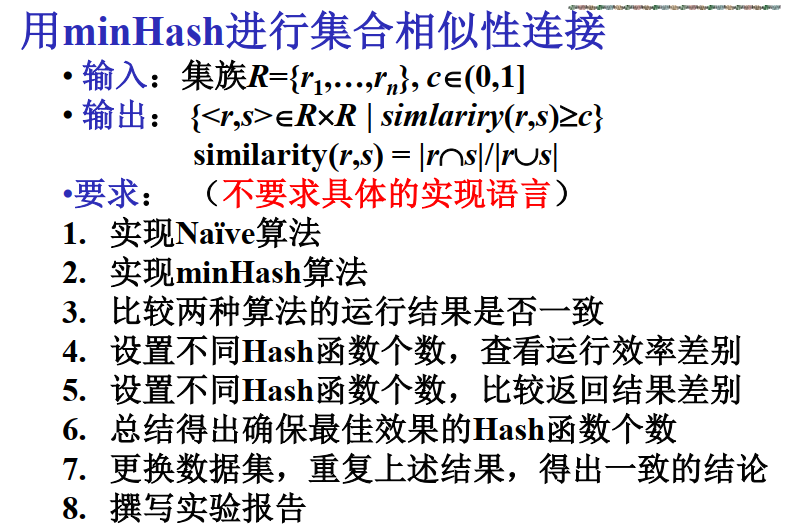
学号：1180301005

班级：1803101

评分表：（由老师填写）

|  |  |
| --- | --- |
| 最终得分： | |
| 对实验题目的理解是否透彻： | |
| 实验步骤是否完整、可信 ： | |
| 代码质量 ： | |
| 实验报告是否规范 ： | |
| 趣味性、难度加分 ： | |
| 特 色： | 1 |
| 2 |
| 3 |

**一、实验题目概述**



**二、对实验步骤的详细阐述**

1.实现了naïve的集合相似连接方法

2.实现了一个setGroup类，用矩阵形式表示集族，并且支持从文件读入集族等操作

3.实现了一个myHash类，实现了不同种类的hash函数

4.实现了一个用来进行数据生成的类myDataGenerator，用来生成不同分布的集族

5.实现了一个类myLSH，作为所有实现集合相似连接方法的类的基类

6.实现了probHelpers类，在其中编写了使用chernoff不等式对误差进行理论分析预测的方法

7.实现了naiveLSH，使用naïve的复杂度为的方法进行集合相似连接近似计算

8.实现了naiveAmpLSH，使用概率放大技术对naiveLSH进行了改进，使得算法的准确率，召回率显著提升

9.实现了typicalLSH，使用最基本的local sensitive hash数据结构进行了优化，但是很显然，在row\_number的个数很多的时候,虽然typicalLSH造成的假阴性会显著减少,但是比较次数会接近于row\_number倍的naiveLSH的比较次数

10.实现了typicalLSH\_R,稍微该进了一下typicalLSH,使得它的最高比较次数不超过naiveLSH

**三、实验数据**

**1. 实验设置**

**实验环境**：

Python 3.9 第三方库:numpy

**数据**：

自己生成的不同分布,不同数据量的数据,最基本的benchmark数据分布为:

左侧是jaccard值的区间,右侧是集对的占比

[0.00,0.05]: 0.0000 [0.05,0.10]: 0.0000 [0.10,0.15]: 0.0000 [0.15,0.20]: 0.0000

[0.20,0.25]: 0.0000 [0.25,0.30]: 0.0087 [0.30,0.35]: 0.3910 [0.35,0.40]: 0.3632

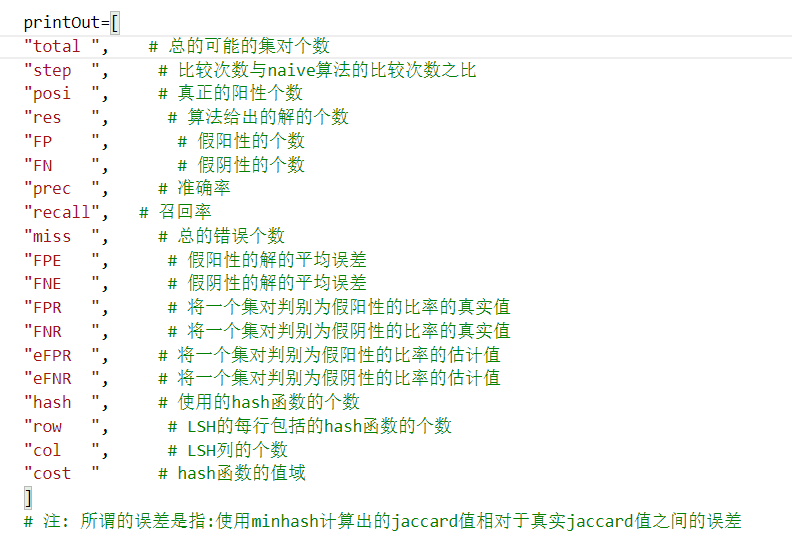
[0.40,0.45]: 0.1103 [0.45,0.50]: 0.0467 [0.50,0.55]: 0.0270 [0.55,0.60]: 0.0174

[0.60,0.65]: 0.0117 [0.65,0.70]: 0.0065 [0.70,0.75]: 0.0046 [0.75,0.80]: 0.0039

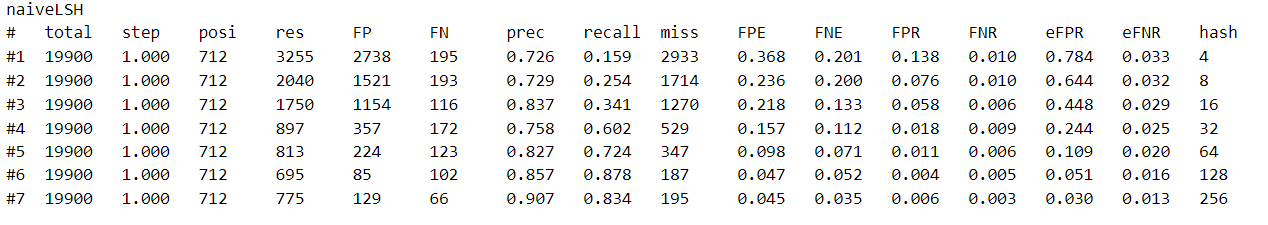
[0.80,0.85]: 0.0025 [0.85,0.90]: 0.0027 [0.90,0.95]: 0.0020 [0.95,1.00]: 0.0018

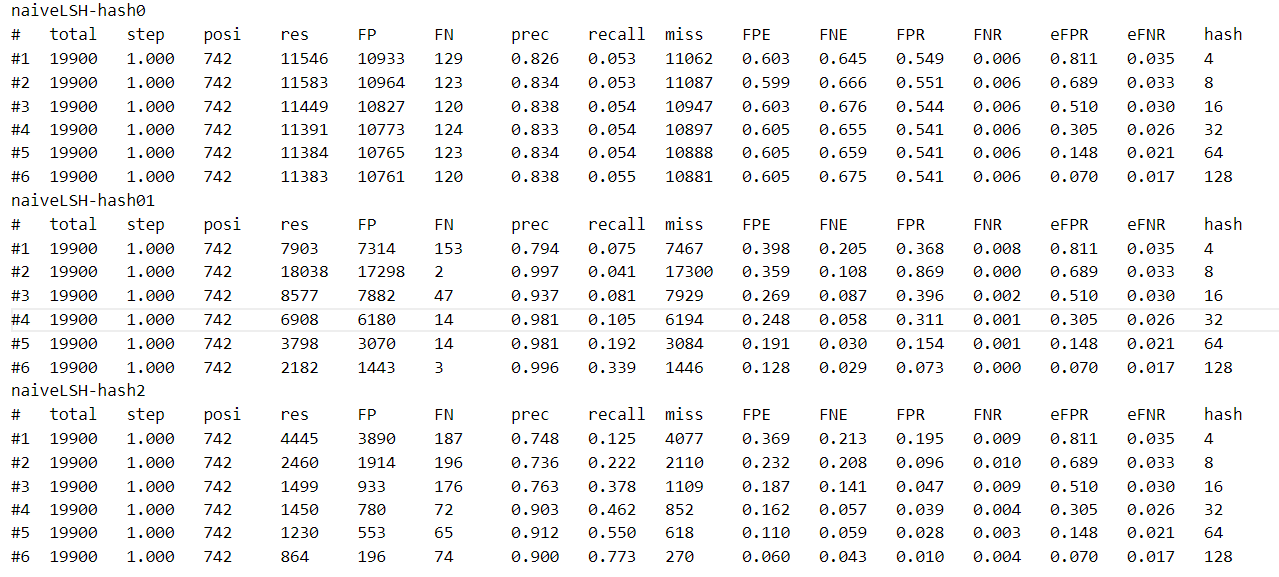
**2. 实验结果**

**简写:**

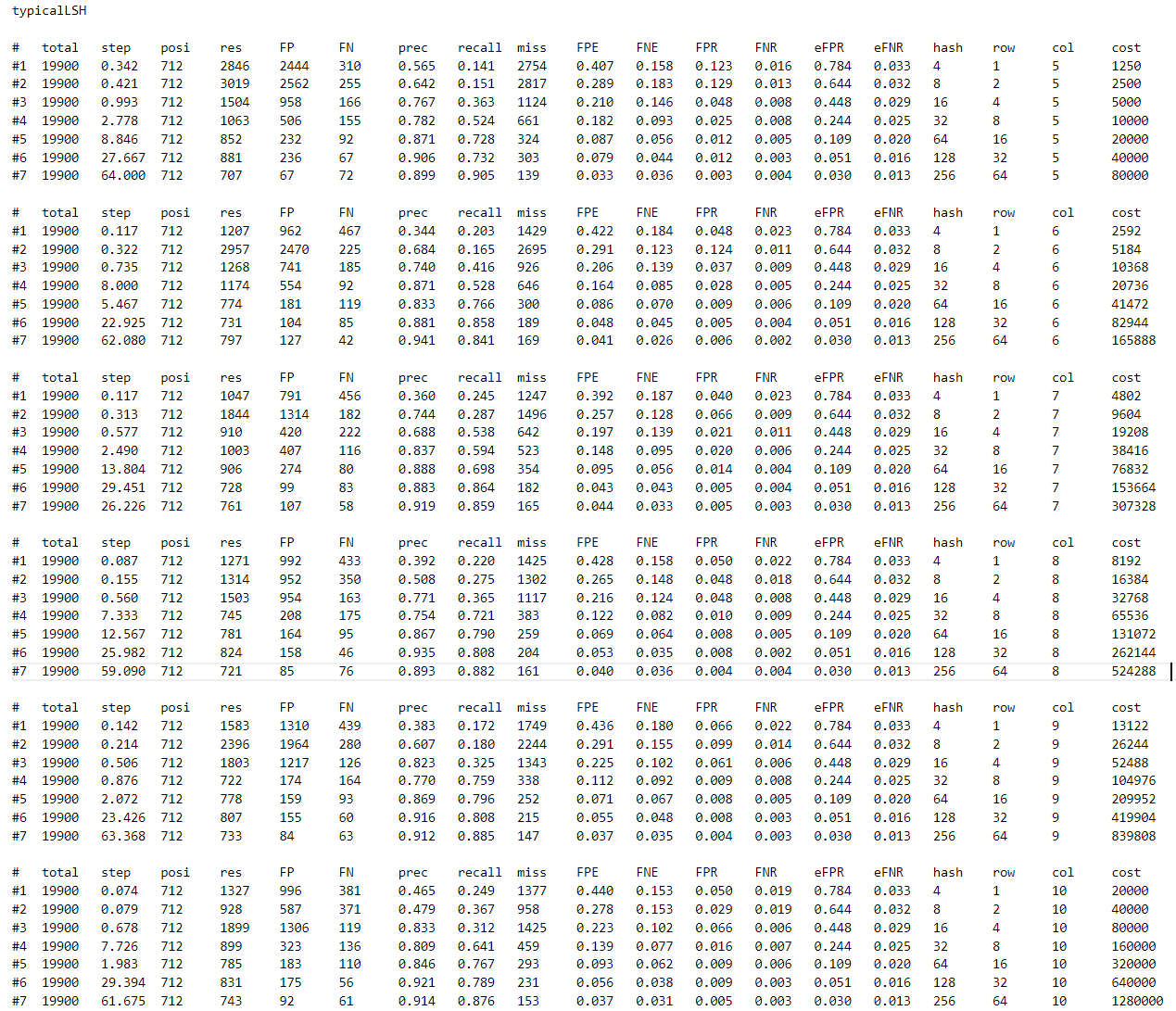
****

**2.1 naiveLSH在不同的hash函数个数下,其准确率和召回率的预测值与真实值**

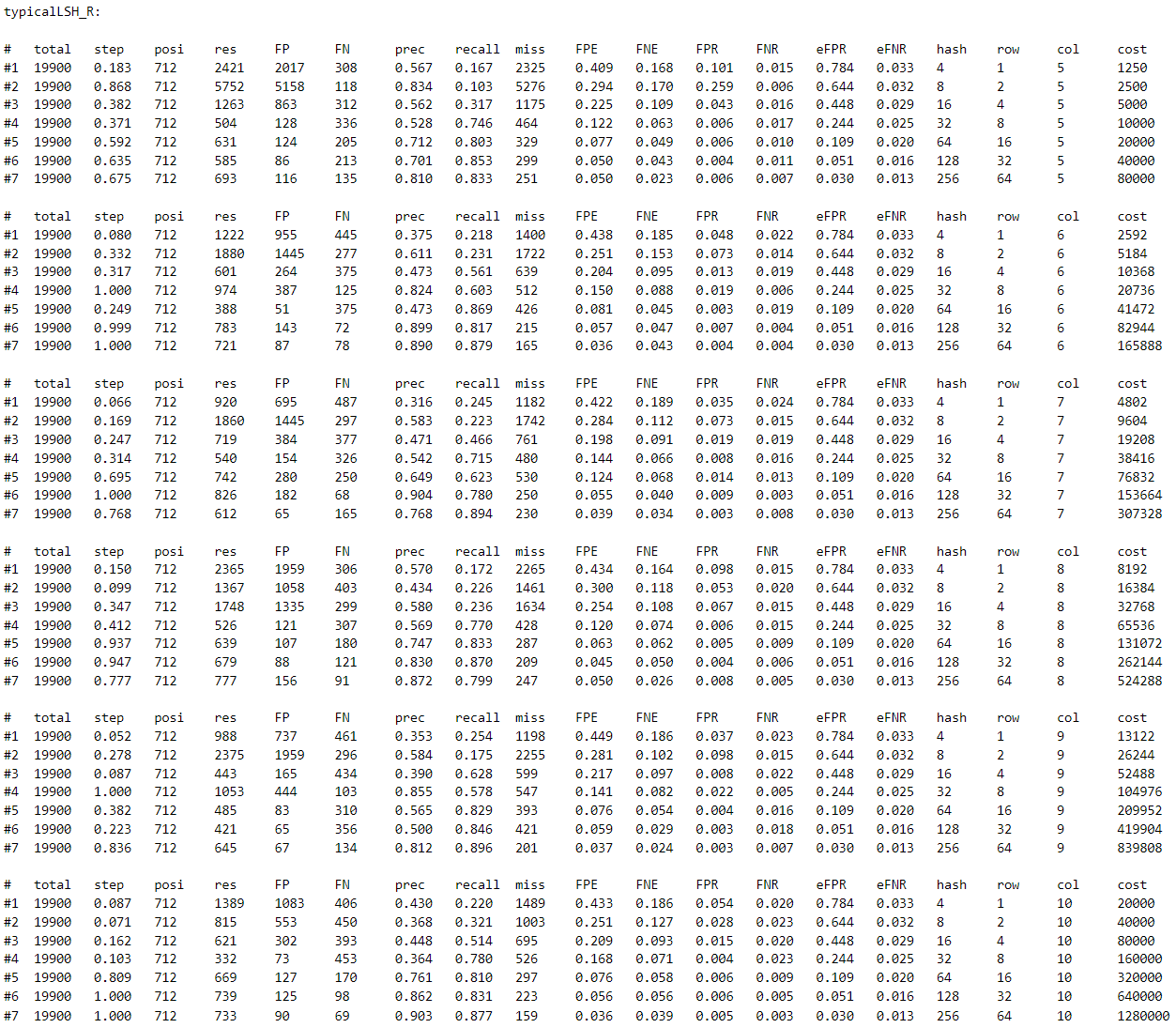
****2.2 naiveLSH在不同种类的hash函数作用下的准确率和召回率



2.3typicalLSH在不同的row\_number以及column\_number下的准确率,召回率以及比较次数



2.4typicalLSH\_R在不同的row\_number以及column\_number下的准确率,召回率以及比较次数



**四、对实验结果的理解和分析**

1.minHash算法的原理:

将两个集合s1,s2使用长度相同的01向量v1,v2来表示,对于第i个二进制比特位,有这样的三种情况:

P1 --> v1[i]=v2[i]=1;

P2 --> v1[i]=v2[i]=0;

P3 --> v1[i]≠v2[i]

显然,s1,s2的jaccard值正好等于.于是,如果能够将v1,v2进行完全随机的打乱再排序,第一次出现非P2情况的概率分布正好是:

Pr[首先出现的是P1]=

Pr[首先出现的是P3]=

因此可以使用hash后的v1,v2中首先出现的是P1还是P3的概率来估计s1,s2的jaccard值

2.minHash的性质以及使用chernoff不等式对minHash性能的估测:

可以将minHash看作是一个N次泊松实验过程,因此可以使用chernoff不等式来分析预测它的效果.

由chernoff不等式得:

[a].假设f为估算后满足c,然而实际上是假阳性的集对,那么将区间[0,c]等分为k份,记为r1,r2,...,rk

[b].通过dis()函数,获得这些区间的占比,并且用区间的中位数代表区间的jaccard值的大小,分别记作d1,d2,...,dk;j1,j2,...,jk

[c].因为从属于第i个区间的集对,如果在估测后显示为假阳性,则估测的误差大小是c-ji,误差的程度是ei=(c-ji)/ji

[d].假设使用的hash函数的个数是N,则根据minHash原理,从属于第i个区间的集对在minHash中相等的hash个数的期望值为N\*ji

[e].根据chernoff不等式,从属于第i个区间的集对被N个minHash估测后,显示为假阳性的概率小于chernoff(N\*ji,ei)[0],记作pi

[f].那么对于任意一个集对pa,Pr[pa被N个minHash估计为假阳性]=

[g].综上所述,Pr[pa被N个minHash估计为假阳性]小于等于

2.1Hash函数个数对minHash准确率的影响:

根据上述分析,在**Hash函数独立性很高**的情况下,显然是Hash函数个数越多,minHash的准确率越高,召回率越高,效果越好,但是复杂度会显然地提升.然而在**Hash函数独立性比较低**,冲撞概率很高的情况下,minHash的假阳性率会极大地提升,增多Hash函数个数也不会明显地降低出错的概率.

2.2Hash函数种类对minHash准确率的影响:

通过实验证明,三种Hash函数中,值域范围大独立性强的Hash函数在minHash估计问题中的估计效果最好.

2.3数据集分布以及c的大小对minHash准确率的影响:

一.通过实验数据还有chernoff分析可知,在标准c邻域中的集对个数对minHash的准确率,召回率有较大的影响,c的值越大,则出错的概率越小

3.LSH的实现与问题

3.1 typicalLSH能够在row\_number较小的情况下降低比较次数,但是在row\_number较大的情况下反而会提高比较次数.

3.2经过改进,typicalLSH\_R可以适应row\_number比较大的情况.

**五、实验过程中最值得说起的几个方面**

1.使用切尔诺夫不等式对minHash的准确性进行预估,从而使用概率放大技术,求解出能够达到某种准确性所需要的最小minHash个数的阈值.

2.使用LSH对minHash方法下的集合相似连接问题进行的优化,并且进行了对优化方法的进一步优化;按照原始方法,row\_number越大则因为LSH而导致的假阴性概率越低,但是会导致时间复杂度趋近于row\_number倍的naïve算法复杂度.改造后的LSH只存储k个索引,不仅减少了空间消耗,而且在k很小的情况下能够保证LSH优化minHash程度更高且完全不会发生退化现象.