大数据技术与应用 - Homework 1

April 15, 2021

要求及说明：

- **诚信代码** 可以分组讨论大作业，但是必须独立完成解答。在提交的书面材料必须明确地写明参与讨论的其他组员名字。请在书面材料的末尾附上以下内容:
  “本人承诺本次作业的解答独立完成，解决过程中曾与 xxx、xxx 等同学共同讨论。”

---

1 Spark (25 分)

编写 PySpark，实现“你可能认识的人”社交网络推荐算法。主要思想是，如果两个人有很多共同好友，那么系统应该推荐再这两个人之间建立联系。

数据

- 作业附带`q1/data` 目录下的`soc-LiveJournal1Adj.txt` 文件
- 文件包含多行邻接列表，格式如下:
  `<User><TAB><Friends>`
  其中`<User>` 代表用户的唯一整数 ID；`<Friends>` 代表用户的朋友，为逗号隔开的整数列表。注意朋友关系是相互的 (即朋友之间的边是无向的)：如果 A 是 B 的朋友，那么 B 也是 A 的朋友。提供的数据中保证如果 A 在 B 的朋友列表中出现，B 也会在 A 的朋友列表里。

算法 使用一个简单的算法：对于每个用户 U，算法会推荐 N=10 个还不是 U 的朋友的用户，但他们与 U 有最多的共同好友。

输出

- 输出应该为每个用户包含一行如下格式
  `<User><TAB><Recommendations>`
  其中`<User>` 代表用户的唯一整数 ID；`<Recommendations>` 是逗号隔开的整数 ID 列表，代表算法推荐的 User 可能认识的人，按照共同好友数量的降序排列。
- 如果用户的 2 度好友 (即上述通过中间好友建立关联的朋友) 少于 10 个，也请将这些推荐按结果按共同好友数量的降序悉数列出。如果用户没有好友，你可以输出一个空的列表。如果在推荐的结果中共同好友的数量相同，请将被推荐好友按他们的 ID 数值升序排列。

流程简述 请用简明的几句话描述你是如何使用 PySpark 解决这个问题的。

提示

- 复用`Colab 0` 中的环境初始化代码
- 逐行检查你每个步骤的输出。比如使用`.take(X)` 方法查看 RDD 的前`X` 个元素
- 为了便于验证，**用户 ID 11** 的推荐结果应该是：
  `27552,7785,27573,27574,27589,27590,27600,27617,27620,27667`

提交内容

(1) 你的代码
(2) 关于 spark 处理本题数据流程的简短描述
(3) 以下用户 ID 的好友推荐列表: 924, 8941, 8942, 9019, 9020, 9021, 9022, 9990, 9992, 9993.

2 关联规则 (30 分)

关联规则经常被零售商用于 Market Basket Analysis (MBA)，分析顾客的购买行为。这部分信息可以进一步被用于商品的交叉销售和追加销售、促销、会员计划、商店设计、折扣计划等其他用途。

项集评价 当你找出数据集中的频繁项集时，你需要选择其中的一个自己作为你的推荐结果。通常用于衡量关联规则重要性和有利性的指标有:

1. **Confidence** (记为 $conf(A \to B)$): *Confidence* 被定义为 basket 中已经包含 A 时，B 出现的概率:
$$conf(A \to B) = Pr(B|A), \tag{1}$$
其中 $Pr(B|A)$ 是 A 出现时 B 也出现的条件概率。
2. **Lift** (记为 $lift(A \to B)$): *Lift* 衡量的是 "A 和 B 一起出现" 的情况比 "A 和 B 相互独立" 的情况多多少:
$$lift(A \to B) = \frac{conf(A \to B)}{S(B)} \tag{2}$$
其中 $S(B) = \frac{Support(B)}{N}$，而 $N = \#baskets$ 即交易总笔数
3. **Conviction** (记为 $conv(A \to B)$): *Conviction* 比较 "如果 A 和 B 相互独立，A 出现时 B 不出现的概率" 和 "实际数据中观察到的 A 出现时 B 不出现" 的概率:
$$conv(A \to B) = \frac{1 - S(B)}{1 - conf(A \to B)} \tag{3}$$

(a) (3 分) *confidence* 的一个缺点是它忽略了 $Pr(B)$。请解释为什么这是一个缺点？以及为什么 *lift* 和 *conviction* 没有这个缺点？

(b) (3 分) 如果一个衡量指标 measure 满足 $measure(A \to B) = measure(B \to A)$，则称 measure 是对称的。以上那些指标是对称的？请给出证明或者反例。

(c) (4 分) 完美蕴含是指总是成立的规则 (即规则相关的条件概率是 1)。如果一个指标 measure 能在所有的完美蕴含下都取到这个指标的最大值，那么称这个指标是值得拥有的，这使得鉴别最好的规则更容易。以上哪些指标符合这个特性？请举例说明。

商品推荐中的应用 将传统商品或服务卖给已有顾客的行为称为交叉销售。提供商品推荐是线上零售商常用交叉销售的一个例子。商品推荐的一种简单方法是推荐经常被顾客一起浏览的商品。

假设我们需要基于顾客的在线浏览记录为他们推荐新的商品。编写一段程序，使用 A-prior 算法来找出经常被一起浏览的商品。将 support 固定为 $s = 100$(即至少出现 100 次可以被认为足够频繁)，找出规模为 2 和 3 的 itemset。

使用目录`q2/data` 目录中的在线浏览是行为数据集`browsing.txt` 文件。每行代表一个用户的一次浏览会话。在每行中 8 个字符代表被浏览的商品 ID，ID 之间用空格分开。

这里提供两条用于辅助检验的结果: (1) 第一遍扫描后得到的频繁项规模为 647($|L_1| = 647$)，(2) 问题 (d) 中的前 5 对结果的 confidence 得分均超过 0.985。

(d) (10 分) 找出所有的二元项集 (X, Y)，满足 $\{X, Y\}$ 的 support 至少为 100。对于所有这些二元项集，计算对应的关联规则 confidence 得分: $X \to Y$ 和 $Y \to X$。将这些规则按照 confidence 得分的降序排列，在书面材料中列出前五条规则。分数相同的情况下，按照关联规则左侧 item ID 的词典顺序排序。

(e) (10 分) 找出所有的三元项集 (X, Y, Z) 可满足 $\{X, Y, Z\}$ 的 support 至少为 100。对于所有的这些三元项集计算对应的关联规则 confidence 得分: $(X, Y) \to Z, (X, Z) \to Y, (Y, Z) \to X$。将这些规则按照 confidence 得分的降序排列，在书面材料中列出前五条规则。

提交内容 提交你的代码以及以下书面材料:

(1) 2(a) 的解释
(2) 2(b) 的证明
(3) 2(c) 的解释
(4) 2(d) 中的前 5 条规则及其得分
(5) 2(e) 中的前 5 条规则及其得分

3 局部敏感哈希 (15 分)

在模拟矩阵中行的随机重排列时，我们可以只对 n 行中随机选中的 k 行哈希来节省时间。这个做法的缺点是当某一列的 k 行都不包含 1 时，min-hash 的结果是“未知”，也就是我们无法找到一个包含 1 的重排行号作为 min-hash 的值。将两个 min-hash 的值都是未知的列当成相似文档是错误的。然而如果得到“未知” min-hash 值的概率很低，我们也可以容忍这种情况，只要在计算两列的 min-hash 的相似比例时将未知 min-hash 忽略掉就行。

在题 (a) 中我们会确定在只用所有 n 行的子集 k 行计算 min-hash 值时得到“未知”的概率上界，在题 (b) 中我们会计算在给定容忍度的条件下使用这个上界确定合适的子集规模 k。

(a) (5 分) 假设一列有 m 个 1 和 $n-m$ 个 0，我们随机地挑选 k 行来计算 min-hash。请证明对这一列计算 min-hash 时得到“未知”值的概率至多是 $(\frac{n-k}{n})^m$。

(b) (5 分) 假设我们想让“未知”的概率至多是 e^{-10}。同时假设 n 和 m 都很大 (但是 n 要比 m 和 k 都大得多)，请简单估计一下使得“未知”概率至多为 e^{-10} 的 k 的值。这个值应该是 n 和 m 的函数。提示:(1) 你可以使用 $(\frac{n-k}{n})^m$ 作为得到“未知”的概率。(2) 对于很大的 x，$(1-\frac{1}{x})^x \approx 1/e$

(c) (5 分) 注意: 题 (c) 与上面两题无关，我们不再随机选择所有行的子集

在进行 min-hash 时，你可能会想不用到所有的重排列来轨迹 Jaccard 相似度。例如，只使用循环重排列，也就是先随机选一行 r 作为重排后的第一行，接着是第 $r+1, r+2$ 行直到最后一行，然后回过头来跟上第一行、第二行直到第 $r-1$ 行。对于 n 行，只有 n 种循环重排列。然而这些重排列不足以正确地估计 Jaccard 相似度。

举一个两列的例子，使得它们只使用循环重排列时 min-hash 值相等的概率和原两列的 Jaccard 相似度不相同。在你的答案中需要提供 (a) 包含两列的矩阵 (这两列代表的集合记为 $S1$ 和 $S2$)，(b)$S1$ 和 $S2$ 的 Jaccard 相似度，(c) 一个随机循环重排列为 $S1$ 和 $S2$ 产生相同 min-hash 值的概率。

提交内容

(1) 3(a) 的证明

(2) 3(b) 的推导和最终结果

(3) 3(c) 中要求的 3 项内容

4 LSH 用于近似近邻搜索 (30 分)

本题中我们研究 LSH 用于近似近邻搜索。

假设我们有包含 n 个点的数据集 $\mathcal{A}$，以及距离函数 $d(\cdot,\cdot)$。令 c 是比 1 大的常数。那么 (c,λ) 近似近邻 (Approximate Near Neighbor, ANN) 问题可以定义如下: 给定查询点 z，假设数据集中有一个点 x 使得 $d(x,z) \le \lambda$，返回数据集中的一个点 x' 使得 $d(x',z) \le c\lambda$ (这个点称为一个 (c,λ)-ANN)。参数 c 在问题中代表最大允许的近似系数。

考虑对距离度量 $d(\cdot,\cdot)$ 关于 $(\lambda, c\lambda, p_1, p_2)$ 敏感的 LSH 函数家族 $\mathcal{H}$。令[1]$\mathcal{G} = \mathcal{H}^k = \{g = (h_1,\ldots,h_k) | h_i \in \mathcal{H}, \forall 1 \le i \le k\}$，其中 $k = log_{1/p_2}(n)$。

考虑以下过程:

1. 选择 $\mathcal{G}$ 的 $L = n^\rho$ 个随机成员 $g_1,\ldots,g_L$，其中 $\rho = \frac{log(1/p_1)}{log(1/p_2)}$
2. 使用所有的 $g_i (1 \le i \le L$ 哈希查询点和所有的数据点
3. 从查询点被哈希到的桶中召回至多[2]$3L$ 个数据点 (随机均等选择)
4. 在阶段 3 中选择的点中，将离查询点最近的点作为 (c,λ)-ANN

问题第一部分的目标是证明这个过程以常数概率得到正确解。

(a) [5 分] 令 $W_j = \{x \in \mathcal{A} | g_j(x) = g_j(z)\} (1 \le j \le L)$ 为被哈希函数 g_j 映射到与查询点 z 同一个桶的数据点 x 的集合。定义 $T = \{x \in \mathcal{A} | d(x,z) > c\lambda\}$。证明:

$$Pr\left[\sum_{j=1}^{L} |T \cap W_j| \ge 3L\right] \le \frac{1}{3}$$

(提示: 马尔可夫不等式)

[1]等式 $\mathcal{G} = \mathcal{H}^k$ 使用逻辑 AND 构造，即 $g(x) = g(y)$ 当且仅当 $\mathcal{G}$ 中所有的 h_i 都有 $h_i(x) = h_i(y)$

[2]如果查询点所在的哈希桶中的数据点少于 $3L$，则全部召回

(b) [5 分] 令 $x^* \in \mathcal{A}$ 为满足 $d(x^*, z) \leq \lambda$ 的一个点。证明:

$$Pr[\forall 1 \leq j \leq L, g_j(x^*) \neq g_j(z)] \leq \frac{1}{e}$$

(c) [5 分] 总结得出算法过程返回的点以大于某个固定常量的概率确实是一个 (c, λ)-ANN

(d) [15 分] 目录`q4/data` 中包含一个图像数据集`patches.csv`
数据集的每行是一个 20×20 的图像块，以 400 维的向量表示。我们将使用 $\mathbf{R}^{400}$ 上的 L_1 距离度量来定义图像的相似度。我们希望比较基于 LSH 的 ANN 搜索与线性搜索之间的性能。本任务中，你需要使用与数据集一同提供的代码。
初始代码`lsh.py` 中使用 TODO 标出了你需要实现的部分。你会用到`lsh_setup` 和`lsh_search` 函数；线性搜索需要你自己实现。本练习中可以为`lsh_setup` 使用默认参数 $L = 10, k = 24$；当然你也可以用其他的参数值，并解释原因。

- 对与第 $100, 200, 300, \ldots, 1000$ 列中的图像块，分别使用 LSH 和线性搜索找出它们的前 3 近邻（这些图像块自身除外）。这两种方法的平均搜索用时是多少?
- 假设 $\{z_j | 1 \leq j \leq 10\}$ 代表上述图像块（即 z_j 是第 $100j$ 列的图像块）,$\{x_{ij}\}_{i=1}^3$ 是通过 LSH 找到的 z_j 的近邻，$\{x_{ij}^*\}_{i=1}^3$ 是通过线性搜索找到的近邻，计算以下误差度量:

$$error = \frac{1}{10} \sum_{j=1}^{10} \frac{\sum_{i=1}^3 d(x_{ij}, z_j)}{\sum_{i=1}^3 d(x_{ij}^*, z_j)}$$

  将误差值绘制为关于 L 的函数 (取 $L = 10, 12, 14, \ldots, 20$，此时固定 $k = 24$)。类似地，也将误差绘制为关于 k 的函数 (取 $k = 16, 18, 20, 22, 24$，此时固定 $L = 10$)。简要地描述这两个函数绘图。
- 最后，画出各前 10 个使用两种搜索方法找到的第 100 列中图像块最近邻（LSH 使用默认参数 $L = 10, k = 24$，或者你的自选参数)。视觉上这两种结果看起来如何?（你需要用到`plot` 函数)

提交内容

(i) 4(a) 的证明

(ii) 4(b) 的证明

(iii) 4(c) 中为什么过程返回的点确实是一个 (c, λ)-ANN

(iv) 4(d) 中

- LSH 和线性搜索的平均时间
- 误差值作为关于 L 和 K 函数的两个绘图，以及这两个绘图的简要描述
- 画出两种方法找出的 10 个最近邻（包括原图本身)，以及简要的视觉比较描述

(v) 4(d) 的代码