【RS】协同过滤-基础篇

原创 机智的叉烧 CS的陋室 2019-02-20

Xenogenesis

TheFatRat - Xenogenesis

【RS】

本栏目是结合我最近上的七月在线的课、自己自学、以及一些个人的经验推出的专栏，从推荐系统的基础到一些比较好的case，我都会总结发布，当然，按照我往期的风格，更加倾向于去讨论一些网上其实讲得不够的东西，非常推荐大家能多看看并且讨论，欢迎大家给出宝贵意见，觉得不错请点击推文最后的好看，感谢各位的支持。

往期回顾：

- 【NLP.TM】GloVe模型及其Python实现
- 【陋室推荐】| 2018-2-15
- 技术向：推荐学习推荐系统（深度思考，不是广告）
- 【RS】推荐系统的评估

协同过滤（Collaborative Filtering，CF）是推荐系统中最为基础的算法，目前主要被用在召回中，当然也可以使一些规模比较小的推荐系统的核心算法，在[1]中也被成为基于邻域的算法。下面我分4篇来详细对协同过滤方法进行讲解，第一篇主要讲协同过滤的基础，第二篇详细讲基于用户的协同过滤，第三篇讲基于物品的协同过滤，第四篇则进行一定的总结，以及提出一些常见的改进方案，供大家参考。

第一篇，谈协同过滤的思想，个人理解，大段文字预警，没有代码，如果急求代码和实现的话看到这里可以先不看了，直接看下一篇就行，user-based的代码其实差不多写完了：https://gitee.com/chashaozgr/noteLibrary/blob/master/rs/src/user_CF，，但是文案还没写完，不过个人感觉，**理解算法本身的含义和思想远比代码本身来的重要。**。

懒人目录

- 协同过滤的思想
- 协同过滤第一步：相似
- 协同过滤第二步：推断
- 小结

协同过滤的思想

之所以想单独花一篇写基础，原因是发现很多文章，包括比较著名的[1]在内，都是直接上来就是讲user-based和item-based，这样就忽略了协同过滤本身的含义，所以这里想展开讨论协同过滤的思想，将算法本身进一步抽象，以便在更多的领域进行应用，而不是简单的局限在推荐系统本身。当然，只是个人的理解，有错误的欢迎讨论。

说到推荐，其实就是去构建一个“用户-物品”的关系，目标是对用户需要的、商家渴望推广的商品根据一定的规则进行匹配，抽象的，其实就是构建一个函数关系，去衡量任意一个用户和物品之间的关联度。

协同过滤的主要思想就是“利用兴趣相同、喜好相似的群体来为用户推荐产品”，在数学上，其实可以理解为，**对一个矩阵M（这里就是“用户-物品”相关性矩阵），已知上面部分值（已有的用户喜好程度，如打分等），去推测（填写）相应空白处的值。**而填写的方法（例如坐标为(X,Y)的值），就是找到相似的多个X'，根据这些X'对Y的喜好程度推断，或者是根据X对与Y相似的Y'的喜好程度来推断。因此，协同过滤就分为两个大步走，一个是**相似**，一个是**推断**。

这里用了一个很有意思的思想，就是“**转化**”，转化说着容易，但是想着却并不容易，协同过滤在这里看来，就是在做一个转化，用户和物品的关联不好比，那我就将物品转化为用户组合的形式，然后将用户和用户组合进行对比（user-based CF），或者是将用户转化为物品组合，然后将物品和物品组合进行对比（item-based CF），从而将两者构成联系，这是CF的精妙之处，**这种方式的精妙，令其使用非常少的信息（没有用到用户性质和物品性质而直接只是两者的交互行为）就能有相对较好的推荐结果。**

道理都懂，但是能不能想到就是另一回事了，这种“转化”，轻松直接地构建了用户和物品的关系，同时具有很强的解释性，这是很多深度学习方法所没有的。因此用作召回可以保证推荐出来的东西不会太过奇怪，感觉是做了一个**兜底**。

有关user-based CF和item-based CF不在这里展开讲，后面会分两篇展开讲原理和实现，这里继续讲一些抽象，以及**user-based CF和item-based CF**都用到的思想。协同过滤分两步走，一个是**相似**，一个是**推断**，因此这里我也分这两块讲。

相似

相似是一个非常常见的问题，之前我的公众号也讨论过这个问题，衡量两者的相似性，其实就是衡量两者的差距有多大，即“距离”，这里其实就是计算距离的问题，差的只是在实际中用什么数据来衡量距离了，那么在推荐系统，实质上就是从喜好程度来衡量距离，换言之，**两个人共同喜欢的东西越多，那么两个人就越相似（user-based）**，或者是**两个物品共同喜欢的人越多，这两个物品就越相似（item-based）**，如果是一个“非黑即白”的二分类问题，那很简单，就是一个共现问题了，这个在[1]和[2]中都有提到，然而如果是一个打分问题，例如0-5分，那就是一个非常灵活的距离问题了。

对于共现问题，通过集合的形式去计算就很清楚。

$$d(i,j) = \frac{(|N_i \cap N_j|)}{(|N_i \cup N_j|)}$$

i和j的距离，就是两者共同喜欢占两者之一喜欢的比例，之所以先讲这个，因为这个是满足“距离”所要求的自反性，即i与j的距离要和j与i的距离相等，但是实际上，在推荐系统中，其实衡量i和j的关系，只需要考虑i本身即可，并不要求了解j喜欢而i不喜欢的东西，因此又可以改进为：

$$d(i,j) = \frac{(|N_i \cap N_j|)}{(|N_i|)}$$

而在更为灵活的距离问题中，则要考虑连续型的关系了，因此在这里抛开例如只有0,1,2,3,4分之类的判断，而直接考虑连续型的问题，在推荐系统中，一般采用这两种衡量距离的方法：**欧氏距离和余弦距离**（有的地方叫皮尔逊相关系数，大家会发现在实际运用中，其实是一样的）。

$$d(i,j) = \frac{\sqrt{\sum_{k \in N_i \cap N_j} (S_{k,i} - S_{k,j})^2}}{(|N_i \cap N_j|)}$$

$$d(i,j) = \frac{\sum_{k \in N_i \cap N_j} (S_{k,i} \times S_{k,j})}{\sqrt{\sum_{k \in N_i \cap N_j} S_{k,i}^2} \times \sqrt{\sum_{k \in N_i \cap N_j} S_{k,j}^2}}$$

欧氏距离强调分数本身要足够接近，余弦距离要求分数的趋势接近（把不同人对好东西的打分严格程度不同，例如有人认为5分就是很棒，但是有的人比较严格，只能给4分，其实这个人压根就没给过5分），两者各有优点，在实际场景中，很多情况两者的结果相近，但是还是会有一些区别。

推断

推断在这里实际上可以理解为一个“数学期望”，说人话是一个加权平均，权重就是相似人和目标用户的相似度，平均值是喜好程度的平均值，根据多个相似人的共同喜好程度推断本人的喜好程度，根据排序得到最

终的推荐结果。

$$r(x,y)=\frac{\sum_{x'\in N(x)} sim(x',x)\times y}{\sum_{x'\in N(x)} sim(x',x)}$$

值得注意的是，在上面计算的是距离，距离越大两则越不相近，而在这里sim是指相似度，相似度越大两者越相近，这两者的转化一般用倒数、相反数等方式可以转化，注意一下即可。

如此一来，就得到了用户和物品的关系，排序后即可完成推荐。

小结

有关协同过滤，我就讲这么多，核心要点是这些：

- 抽象的，协同过滤解决的是一个“已知一个相关性矩阵的部分值，推断其他值”的问题。
- 协同过滤在推荐系统中的目标是衡量“用户”和“物品”之间的匹配程度。
- 协同过滤通过“转化”，沟通“用户”和“物品”的关系。
- 协同过滤分两步走，“相似”和“推断”。

参考文献

[1] 项亮，《推荐系统实践》

[2] 黄昕，赵伟，王本友等，《推荐系统与深度学习》（这本书强烈推荐）