Item Based CF

## Item-Based CF

假设：用户喜欢跟他过去喜欢的物品相似的物品，历史上相似的物品在未来也相似

方法：给定用户u，找到他过去喜欢的物品的集合R(u)，把和R(u)相似的物品推荐给u

基于物品的协同过滤，以物品为核心，它是对基于用户的协同过滤的一种改良。

理解为：“买了这款商品的用户，还买过什么。”

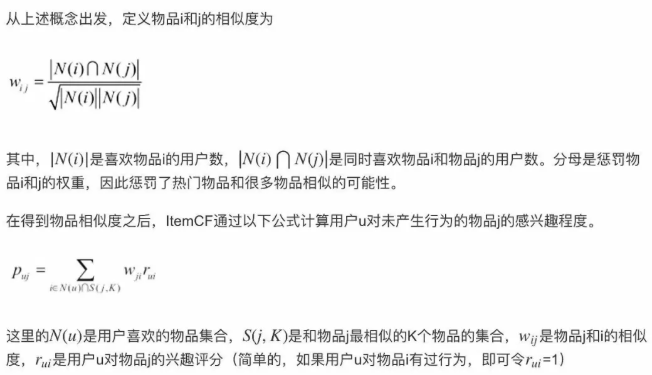
======================================

Item-CF是目前应用最广泛的召回算法，其原理是根据两个商品被同时点击的频率来计算两个商品之间的相似度 simScore，得到 i2i 表；然后通过用户的 trigger 去查询 i2i 表，扩展用户感兴趣的商品。Item-CF 的基本算法虽然简单，但是要获得更好的效果，往往需要根据实际的业务场景进行调优。清除爬虫、刷单等噪声数据，合理选择计算商品之间相似度的数据的时间窗口，引入时间衰减，只考虑同一个类目下商品对，归一化、截断、打散等策略对优化 Item-CF 的效果都有很大的帮助。

===================================================

Item-CF：基于用户对物品的偏好找到相似的物品，然后根据用户的历史偏好，推荐相似的物品给他。（假设用户会喜欢和他以前喜欢的东西相似的东西）

给用户推荐那些和他们之前喜欢的物品相似的物品。不过Item-CF不是利用物品的内容计算物品之间相似度，而是利用用户的行为记录（例如，最开头提到的天猫根据两个商品被同时点击的频率来计算两个商品之间的相似度，本文我们将根据用户购买商品行为来计算两个商品之前的相似度）。



【步骤】

1、将物品的用户当作物品的特征向量，然后计算物品产品之间的相似度，得到物品相似度矩阵。

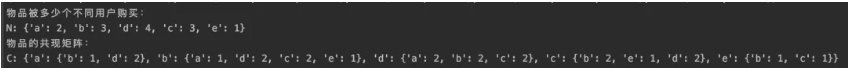
2、从用户已经产生行为的物品中找到于其相似的K个物品（从相似度矩阵中），两次加权累和（用户产生行为物品的评分\*相似物品相似度），找出评分最高的N件物品推荐给用户。

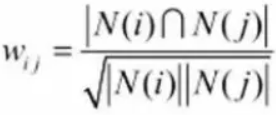
【举例】

1、收集用户数据

例如：用户A购买物品a b d，用户B购买物品b c e，用户C购买物品c d，用户D购买物品b c d，用户E购买物品a d。

2、找出物品相似度

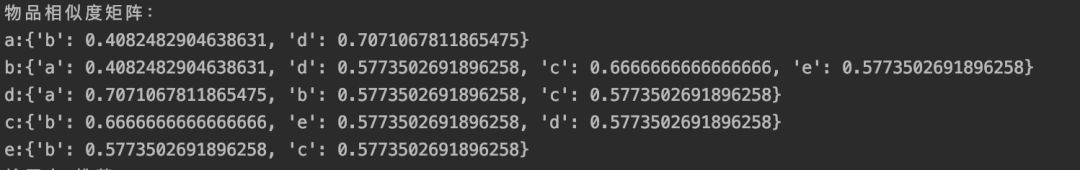




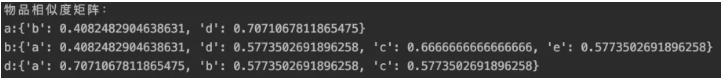
转换公式即：



结果为：



用户A购买过的商品（a b c）的相似度矩阵



进行K=3 排序

a:{'d': 0.7071067811865475, 'b': 0.4082482904638631}  
b:{'c': 0.6666666666666666, 'd': 0.5773502691896258, 'e': 0.5773502691896258}  
d:{'a': 0.7071067811865475, 'b': 0.5773502691896258, 'c': 0.5773502691896258}

3、找出评分最高的N=10件物品推荐给用户

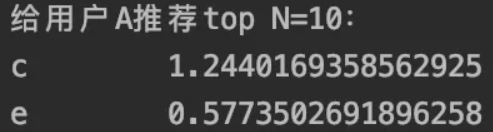


去掉用户已购买的a b d，计算c 和 d :

c: 0.66+ 0.57 = 1.23

e:0.57

排序取top N=10 推荐给用户:



======================================

著名的电子商务公司亚马逊提出了基于物品的协同过滤算法。基于物品的协同过滤算法（简称ItemCF）给用户推荐那些和他们之前喜欢的物品相似的物品。不过，ItemCF算法并不利用物品的内容属性计算物品之间的相似度，它主要通过分析用户的行为记录计算物品之间的相似度。该算法认为，物品A和物品B有很大的相似度是因为喜欢物品A的用户大都也喜欢物品B。基于物品的协同过滤算法可以利用用户的历史行为给推荐结果提供推荐解释，比如给用户推荐《天龙八部》的解释可以是因为用户之前喜欢《射雕英雄传》。亚马逊显示相关物品推荐时的标题是“Customers Who Bought This Item Also Bought”（购买了该商品的用户也经常购买的其他商品）。基于物品的协同过滤算法主要包括以下4个步骤：

（1）构建User-Item矩阵；

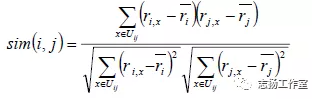
（2）根据UI矩阵来计算列（物品维度）的相似度；

（3）选择特定物品最相似的k个物品；

（4）推荐给特定用户还没有发生过行为的项目。

======================================

相似度（其实就是余弦相似度）

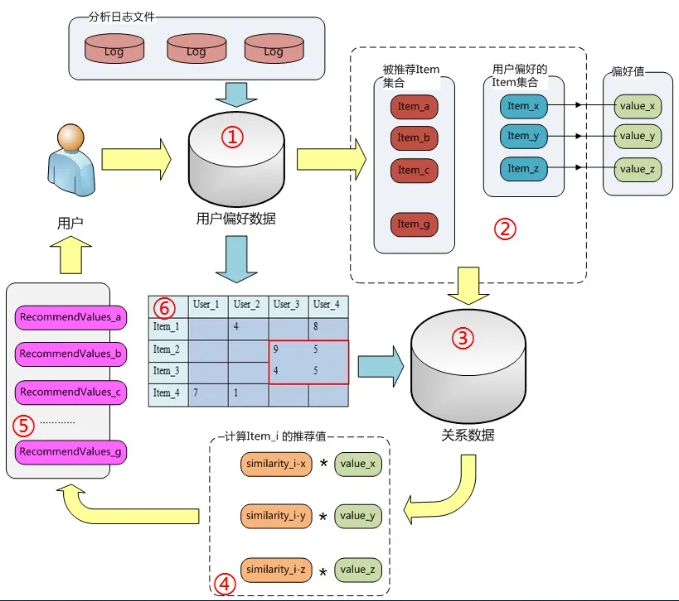


该公式要计算物品i和物品j之间的相似度, ri头上有一杠的代表物品i所有评分的平均分, 之所以要减去平均分是因为有的用户打分严有的松，归一化用户打分避免相互影响。

================================

跟你喜欢的物品相似的物品，你也很有可能喜欢 --基于物品的协同过滤推荐（Item-based CF）

================================================



1. 查找这个用户喜欢过的物品（即偏好的产品，并查出偏好值后面会用），以及还没有喜欢过的商品，前者是推荐运算的根据，后者作为一个产生推荐的一个集合；如②画的那样。
2. 这里是一个可扩展的地方；因为这两部分的数据的作用非常明显，修改这两个集合对后面产生的推荐结果可产生非常直观的影响，比如清洗过滤，或根据用户属性缩小集合；不仅使后面推荐效果更优，运算性能也可以大幅度提高。
3. 查找这两个集合之间的关系，这是一对多的关系：一个没有偏好过的物品与该用户所有偏好过的物品间的关系，有一个值来衡量这个关系叫相似度Similarity；这个关系怎么来的，看蓝色箭头的指向。步骤6
4. 得到这个一对多的关系后，就可以计算这个物品对于这个用户的推荐值了，图中similarity\_i-x表示Item\_i 与 Item\_x 之间的相似度，Item\_x是该用户偏好过得，该用户对其偏好值记为 value\_x ，相乘；Item\_i 与 该用户偏好过的所有物品以此做以上运算后，得到的值取平均值便是 Item\_i的推荐值了。注：有可能Item\_i 不是与所有该用户偏好过的物品都都存在相似性，不存在的，不计算即可；另外这里方便理解介绍的都是最简单的实现；你也可以考一些复杂的数学元素，比如方差来判断离散性等。
5. 这步就简单多了，刚才对该用户没有偏好过的集合中的所有Item都计算了推荐值，这里就会得到一个list，按推荐值由大到小排序，返回前面的一个子集即可。
6. 前面已经提到，关系数据时怎么来的，也是根据用户的偏好数据；你把其看成一个矩阵，横着看过来，参考两个Item间的共同用户，以及共同用户的偏好的值的接近度；这里的可选择的相似度算法很多。

==========================

a.假设用户喜欢跟他过去喜欢的物品相似的物品

b.假设历史上相似的物品在未来也相似

c.给定用户u，找到他过去喜欢的物品的集合R(u)，把和R(u)相似的物品推荐给u.

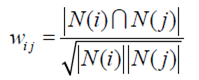
=====================================

3.1ItemCF算法原理

ItemCF算法核心主要分两步：

1. 计算物品之间的相似度：
2. 根据用户的历史行为和物品的相似度，为用户生成推荐列表，排序。

第一步：计算物品之间的相似度；



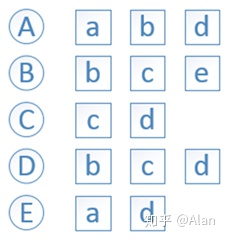
其中，|N(i)|是喜欢物品i的用户数，|N(j)|是喜欢物品j的用户数，|N(i)&N(j)|是同时喜欢物品i和物品j的用户数。

从上面的定义看出，在协同过滤中两个物品产生相似度是因为它们共同被很多用户喜欢，两个物品相似度越高，说明这两个物品共同被很多人喜欢。

这里面蕴含着一个假设：就是假设每个用户的兴趣都局限在某几个方面，因此如果两个物品属于一个用户的兴趣列表，那么这两个物品可能就属于有限的几个领域，而如果两个物品属于很多用户的兴趣列表，那么它们就可能属于同一个领域，因而有很大的相似度。

(用户活跃度大的用户可能喜欢列表中有很多商品，会影响ItemCf算法计算结果准确性，后面优化算法会增加一个用户活跃度分子)

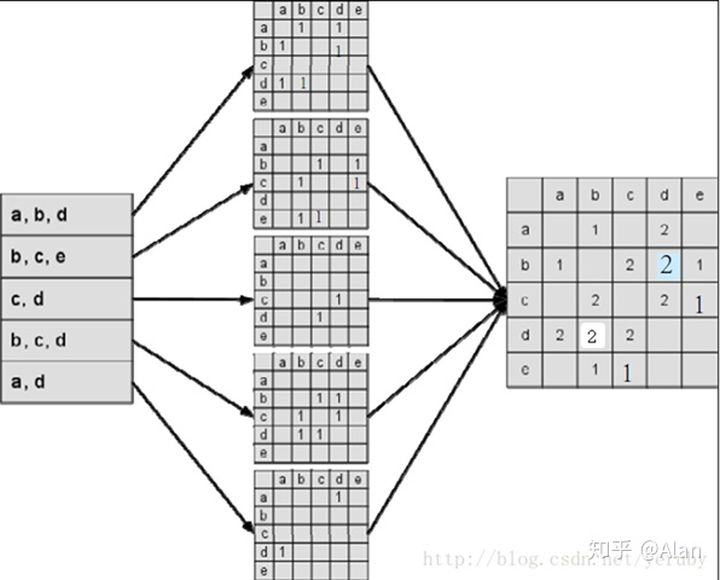
举例，用户A对物品a、b、d有过行为，用户B对物品b、c、e有过行为，等等；



依此构建用户——物品倒排表：物品a被用户A、E有过行为，等等；



建立物品相似度矩阵C：

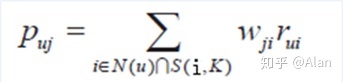


其中，C[i][j]记录了同时喜欢物品i和物品j的用户数，这样我们就可以得到物品之间的相似度矩阵W。

在得到物品之间的相似度后，进入第二步。

第二步：根据物品的相似度和用户的历史行为给用户生成推荐列表；

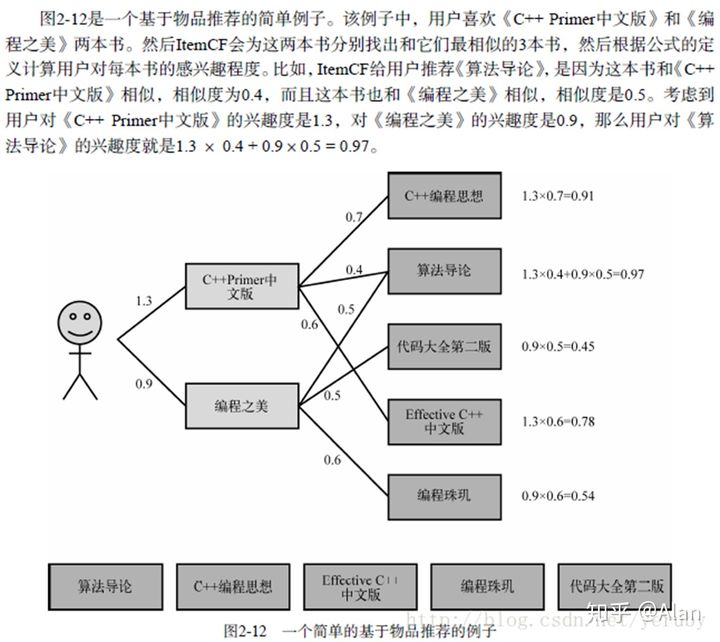
ItemCF通过如下公式计算用户u对一个物品j的兴趣：



其中，Puj表示用户u对物品j的兴趣，N(u)表示用户喜欢的物品集合（i是该用户喜欢的某一个物品），S(i,k)表示和物品i最相似的K个物品集合（j是这个集合中的某一个物品），Wji表示物品j和物品i的相似度，Rui表示用户u对物品i的兴趣（这里简化Rui都等于1）。

该公式的含义是：和用户历史上感兴趣的物品越相似的物品，越有可能在用户的推荐列表中获得比较高的排名。

下面是一个书中的例子，帮助理解ItemCF过程：

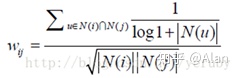


3.3物品协同算法（ItemCF）的优化

下面是提到的几个优化方法：

（1）用户活跃度对物品相似度的影响

即认为活跃用户对物品相似度的贡献应该小于不活跃的用户，所以增加一个IUF（Inverse User Frequence）参数来修正物品相似度的计算公式：

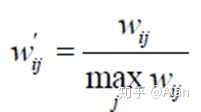


用这种相似度计算的ItemCF被记为ItemCF-IUF。

ItemCF-IUF在准确率和召回率两个指标上和ItemCF相近，但它明显提高了推荐结果的覆盖率，降低了推荐结果的流行度，从这个意义上说，ItemCF-IUF确实改进了ItemCF的综合性能。

（2）物品相似度的归一化

Karypis在研究中发现如果将ItemCF的相似度矩阵按最大值归一化，可以提高推荐的准确度。其研究表明，如果已经得到了物品相似度矩阵w，那么可用如下公式得到归一化之后的相似度矩阵w'：

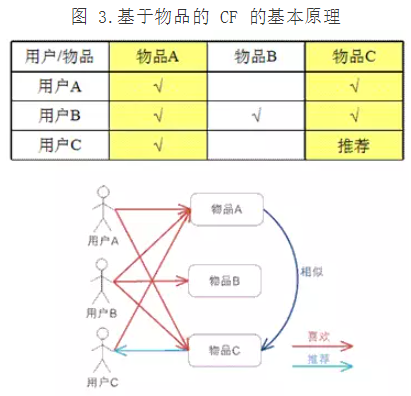


最终结果表明，归一化的好处不仅仅在于增加推荐的准确度，它还可以提高推荐的覆盖率和多样性。

用这种相似度计算的ItemCF被记为ItemCF-Norm。

======================================

基于物品的 CF 的原理和基于用户的 CF 类似，只是在计算邻居时采用物品本身，而不是从用户的角度，即基于用户对物品的偏好找到相似的物品，然后根据用户的历史偏好，推荐相似的物品给他。从计算的角度看，就是将所有用户对某个物品的偏好作为一个向量来计算物品之间的相似度，得到物品的相似物品后，根据用户历史的偏好预测当前用户还没有表示偏好的物品，计算得到一个排序的物品列表作为推荐。图 3 给出了一个例子，对于物品 A，根据所有用户的历史偏好，喜欢物品 A 的用户都喜欢物品 C，得出物品 A 和物品 C 比较相似，而用户 C 喜欢物品 A，那么可以推断出用户 C 可能也喜欢物品 C。

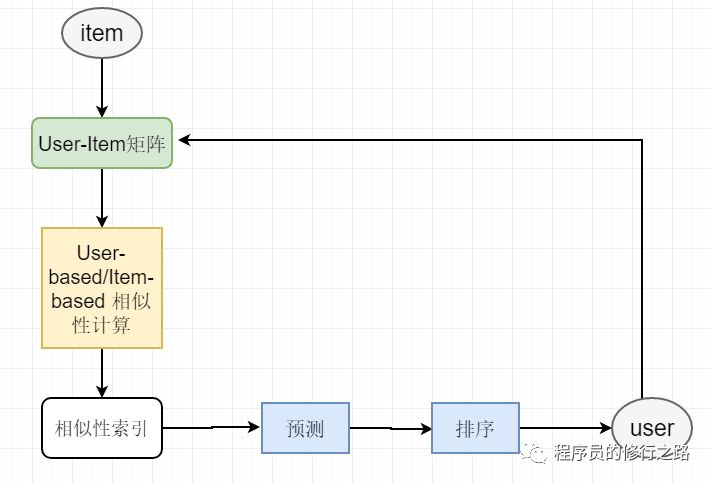


==========================

## Item-Based CF

假设：用户喜欢跟他过去喜欢的物品相似的物品，历史上相似的物品在未来也相似

方法：给定用户u，找到他过去喜欢的物品的集合R(u)，把和R(u)相似的物品推荐给u



a.假设用户喜欢跟他过去喜欢的物品相似的物品

b.假设历史上相似的物品在未来也相似

c.给定用户u，找到他过去喜欢的物品的集合R(u)，把和R(u)相似的物品推荐给u.

# 处理过程

1、数据准备

用户user\_id,物品item\_id，打分score（score可以是用户对某件物品的评分，或是根据用户行为计算出的偏好度得分，比如曝光、点击、收藏的加权得分，具体权重可以参考漏斗模型），数据如下：

| user\_id | item\_id | score |

| ----- | ------------- | ---- |

| id1 | item1 | 3 |

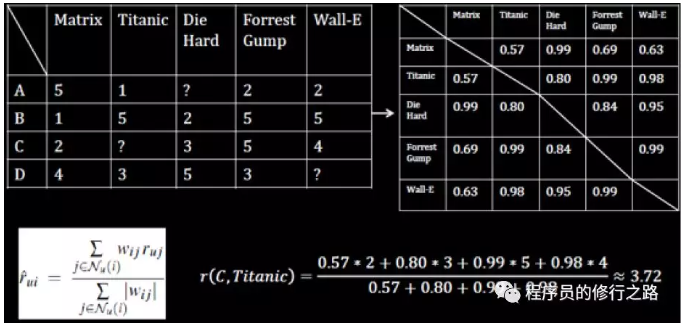
| id1 | item2 | 2 |

| id2 | item1 | 4 |

| id2 | item2 | 3 |

2、计算相似性矩阵

CF算法的关键在于计算获得user或item的相似度矩阵，即UU矩阵和II矩阵。



计算各个Item之间的相似度矩阵，即对两个Item都打过分的id的打分情况作为向量，同理得到item的相似度矩阵。

3、推荐

根据相似度矩阵，选择与目标用户相似度最高的几位用户，在第一张表中选取各自打分较高的物品，形成一个推荐候选集合，准备推荐给目标用户。

# 参考

<https://mp.weixin.qq.com/s/RkizBXpK2CvBlwLUwaKhHg>

<https://mp.weixin.qq.com/s/rvzGQShfEU-waaFLPpRvGg>

<https://mp.weixin.qq.com/s/7SS8fBFkPB7lLnyELMf2jw>

<https://mp.weixin.qq.com/s/S3N9KFu1PXOQ67i2ge8GkQ>

<https://mp.weixin.qq.com/s/lfkTryraR22INdPzR0sidg>

<https://mp.weixin.qq.com/s/kDL7D_5NwzqprUphByKi_w>

<https://mp.weixin.qq.com/s/zvJX9k3u3f_WdtII-QF-aw>

<https://mp.weixin.qq.com/s/lWCR9CizwiqGqSrH2ISxYg>

<https://mp.weixin.qq.com/s/FKkE_wm4O4bYD3MycBFaDQ>

<https://zhuanlan.zhihu.com/p/94024379>

<https://mp.weixin.qq.com/s?__biz=MzAxNDkzNjkzMA==&mid=2247485468&idx=1&sn=dded415b69aa0ef548a87ef582566213&chksm=9b8af6>

<https://mp.weixin.qq.com/s/lNGRz0uzmKj0J3oX6egQ3A>

<https://mp.weixin.qq.com/s/N21N7EYh4xXYSlc1KwEspw>

<https://mp.weixin.qq.com/s/2nXYNT1FQSgxLQ1HM2DJhg>