CF——推荐算法

原创 越前浩波 浩波的笔记 2020-08-27

推荐算法首先要介绍的一定是协同过滤算法了(collaborative filtering，CF)，CF算法的汇总的是所有的<user,item>行为对，有点像朋友推荐，比如用户A和用户B都喜欢差不多的东西(item相似)，用户B喜欢某样东西，但是用户A还没有喜欢，那么此时就将用户B喜欢的item推荐给用户A。

(User-Based CF)，还有一种协同推荐，即对比数据(item)，发现itemA和itemB类似(即被差不多的users喜欢)，就把某user的所有喜欢的item的类似item过滤出来作为候选推荐给该user。

算法的优势：

- 经常能推荐出一些意想不到的结果
- 进行有效的长尾item
- 只依赖用户行为，无需对内容进行深入了解，适用范围广。

算法的劣势：

- 一开始需要大量的<user,item>行为数据，即需要大量冷启动数据
- 很难给出合理的推荐解释

协同过滤的实现方式：

1.基于领域的协同过滤算法

主要是利用<user,item>打分矩阵，利用统计信息来计算user-user，item-item之间的相似度。利用相似度进行排序，得出最终的推荐结果。

- 1.User-Based CF

$$\mathrm{sim}(i,j)=\frac{\sum_{x\in I_{ij}}(R_{i,x}-\bar{R}_i)(R_{j,x}-\bar{R}_j)}{\sqrt{\sum_{x\in I_{ij}}(R_{i,x}-\bar{R}_i)^2}\sqrt{\sum_{x\in I_{ij}}(R_{j,x}-\bar{R}_j)^2}}$$ (皮尔逊相关系数)

计算用户i和用户j之间的相似度， $I(ij)$ 是代表用户i和用户j共同评价过的物品，$R(i,x)$代表用户i对物品x的评分，$\bar{R}_i$代表用户i评分的平均分，之所以减去平均分是因为用户评分标准有差异，归一化以避免差异影响。

但是此公式有欠缺的地方，即没有考虑到热门商品可能会被很多用户所喜欢，需要优化加权重 2 Item-Based CF$\text{sim}(i,j)=\frac{\sum_{x\in U_{ij}}(r_{i,x}-\bar{r}_i)(r_{j,x}-\bar{r}_j)}{\sqrt{\sum_{x\in U_{ij}}(r_{i,x}-\bar{r}_i)^2\sum_{x\in U_{ij}}(r_{j,x}-r_j)^2}}$ (皮尔逊相关系数)

2.基于隐语义的协同过滤算法：

基于隐语义的方法不依赖于共同评分，基本思想是将用户和物品分别映射到某种真实含义未知的特征向量。

比如可以将user的兴趣表示为4维：[颜色偏好,重量要求,设计风格偏好,价格偏好]，item相应的特点表示为4维：[颜色,重量,设计风格,价格]。通过特征向量的内积来判断用户对一个物品的喜好程度。模型会通过最小化损失来学习这两个向量，虽然此方法不要求共同评分，但是还是会面临很大的数据稀疏问题。(NLP技术的LSA和LDA）**3.基于矩阵分解的协同过滤算法：**

SVD分解$R_{U\times I}=P_{U\times K}Q_{K\times I}$(满秩分解，K=Rank(R))

R为user-item评分矩阵，U表示用户数，I表示商品数，利用R在已知评分训练P和Q使得P和Q相乘的结果最好地拟合已知评分，未知评分也就可以用拟合好的PxQ得到。

那么如何进行训练呢？

假设已知评分为$r_{ui},e_{ui}=r_{ui}-\tilde{r}_{ui}$

总误差平方和：$SSE=\sum_{u,i}e_{ui}^2=\sum_{u,i}\left(r_{ui}-\sum_{k=1}^{K}p_{uk}q_{ki}\right)^2$

目标函数中只有训练误差，就很容易导致过拟合问题,引入两个隐语义矩阵的正则项得到RSVD $SSE=\frac{1}{2}\sum_{u,i}e_{ui}^2+\frac{1}{2}\lambda\sum|p|^2=\frac{1}{2}\sum_{u,i}e_{ui}^2+\frac{1}{2}\lambda\sum_u\sum_{k=0}^{K}p_{uk}^2+\frac{1}{2}\lambda\sum_i\sum_{k=0}^{K}q_{ki}^2$

RSVD的改进

用户对商品的打分不仅取决于用户和商品间的某种关系，还取决于用户和商品独有的性质，将这些性质通过基线评分(baseline estimates)或者说是偏置(bias)来表示。

定义用户u对物品i的评分 r_{ui} 的基线评分b_{ui} : $b_{ui} = \mu + b_u + b_i$

其中b_u ， b_i 是用户u和物品i的观测误差(独特属性)，

$\tilde{r}_{ui} = \mu + b_u + b_i + p_u^T q_i$

u为总的平均分，b_u 为用户 u 的属性值，b_{i}为商品i的属性值，假如这两个变量的SSE式子中同样需要惩罚，

$SSE = \frac{1}{2}\sum_{u,i} e_{ui}^2 + \frac{1}{2}\lambda\sum_u |p_u|^2 + \frac{1}{2}\lambda\sum_i |q_i|^2 + \frac{1}{2}\lambda\sum_u b_u^2 + \frac{1}{2}\lambda\sum_u b_i^2$其中:
$e_{ui} = r_{ui} - \mu - b_u - b_i - \sum_{k=1}^K p_{uk}q_{ki}$

ASVD

全名叫Asymmetric-SVD，即非对称SVD，其式子为：
$\tilde{r}_{ui} = \mu + b_u + b_i + q_i^T\left(|R(u)|^{-\frac{1}{2}}\sum_{j\in R(u)}(r_{uj} - \mu - b_u - b_j)x_j + |N(u)|^{-\frac{1}{2}}\sum_{j\in N(u)} y_j\right)$

R(u)表示用户u评过分的商品集合，N(u)表示用户u浏览过但没有评过分的商品集合，X_j, Y_j是商品的属性。在这个式子中，用户矩阵P不见了，取而代之的是R(u)和N(u)来表示用户属性，这具有合理性，因为用户的行为记录本身就能反应用户的喜好。该模型优势之一是：一个商场或者网站的用户数成千上万甚至过亿，存储用户属性的二维矩阵会占用巨大的存储空间，商品数却没那么多。缺点就是迭代时间长，以时间换空间。

SVD++

引入了隐式的反馈，使用用户历史浏览数据，用户历史评分数据，电影的的历史浏览数据，电影的历史评分数据等作为新的参数。

隐含反馈的原因比较复杂，专门给一部分参数空间去建模，每个item对应一个向量y_i ，通过user隐含反馈过的item的集合来刻画用户的偏好。
$\tilde{r}_{ui} = \mu + b_u + b_i + q_i^T\left(p_u + \frac{1}{\sqrt{|N(u)|}}\sum_{j\in N(u)} y_j\right)$

N(u)用户u行为记录(包含浏览的和评过分的商品集合)。

目标函数为：
$\min\sum_{(u,i)\in K}\left(r_{ui} - \mu - b_i - b_u - q_i^T\left(p_u + |N(u)|^{-\frac{1}{2}}\sum_{j\in N(u)} y_j\right)\right)^2 + \lambda_1\left(\|q_i\|^2 + \|p_u\|^2 + \sum_{j\in N(u)}\|y_i\|^2\right) + \lambda_2\left(b_u^2 + b_i^2\right)$

除了以上的协同过滤方法以外，还有：

- 基于关联算法做协同过滤：Apriori，FP Tree，PrefixSpan
- 基于聚类算法做协同过滤：K-Means，BIRCH，DBSCAN，spectral clustering
- 用分类算法做协同过滤：逻辑回归(相比于SVM解释性强)，朴素贝叶斯
- 用回归算法做协同过滤：Ridge回归，回归树，支持向量机回归
- 用神经网络做协同过滤：限制玻尔兹曼机(RBM)
- 用图模型做协同过滤：SimRank系列，马尔可夫模型算法

未来的新方向：

- 基于集成学习的方法和混合推荐
- 基于矩阵分解的方法：分解机(Factorization Machine)和张量分解(Tensor Factorization)
- 基于深度学习的方法：RBM已经展现出较好的推荐效果，后续的DeepFM

邻居的挑选

1.固定数量邻居(K-neighborhoods or Fix-size neighborhoods)

k紧邻算法，算出相似度之后，选取指定结点周围固定的k个最近的结点作为邻居。这种方法对于孤立点效果不好，当附近没有足够多的比较相似的点，就被迫取一些不太相似的点作为邻居，影响了邻居相似的程度。

2.基于相似度阈值的邻居(Threshold-based neighborhoods)

基于相似度门槛的邻居计算是对邻居的远近进行最大值限制，落在以当前点为中心，距离为K的区域中的所有点作为当前点的邻居。这种方法得到的邻居数不确定，但是相似度不会出现较大的误差，对孤立点的处理也更加优化。

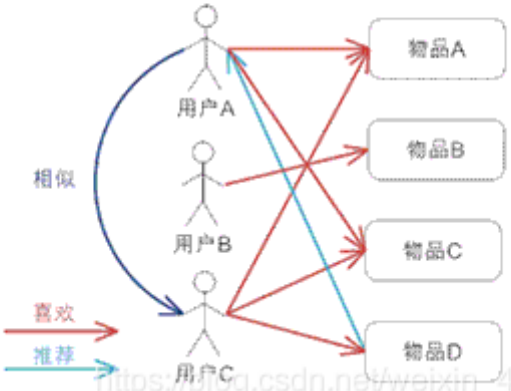
3.计算推荐

User-Based CF 的基本思想相当简单，基于用户对物品的偏好找到相邻邻居用户，然后将邻居用户喜欢的推荐给当前用户。计算上，就是将一个用户对所有物品的偏好作为一个向量来计算用户之间的相似度，找到 K 邻居后，根据邻居的相似度权重以及他们对物品的偏好，预测当前用户没有偏好的未涉及物品，计算得到一个排序的物品列表作为推荐。

Item-Based CF即从物品的角度去考虑，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。

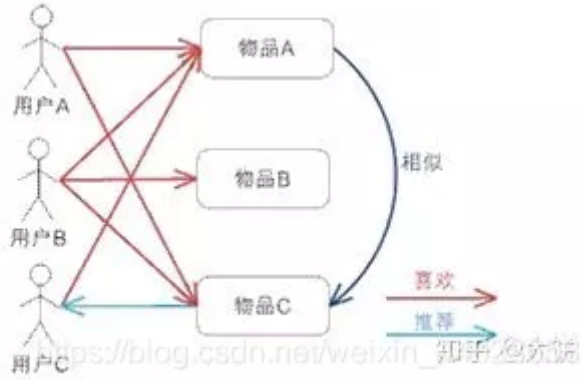
下图给出了一个例子，对于用户 A，根据用户的历史偏好，这里只计算得到一个邻居 - 用户 C，然后将用户 C 喜欢的物品 D 推荐给用户 A。

用户/物品	物品A	物品B	物品C	物品D
用户A	√		√	推荐
用户B		√		
用户C	√		√	√



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。下图给出了一个例子，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。

用户/物品	物品A	物品B	物品C
用户A	√		√
用户B	√	√	√
用户C	√		推荐



在这里插入图片描述

Item-CF和User-CF选择

user和item数量分布以及变化频率

- 如果user数量远远大于item数量, 采用Item-CF效果会更好, 因为同一个item对应的打分会比较多, 而且计算量会相对较少
- 如果item数量远远大于user数量, 则采用User-CF效果会更好, 原因同上
- 在实际生产环境中, 有可能因为用户无登陆, 而cookie信息又极不稳定, 导致只能使用item-cf
- 如果用户行为变化频率很慢(比如小说), 用User-CF结果会比较稳定
- 如果用户行为变化频率很快(比如新闻, 音乐, 电影等), 用Item-CF结果会比较稳定

相关和惊喜的权衡

- item-based出的更偏相关结果, 出的可能都是看起来比较类似的结果
- user-based出的更有可能有惊喜, 因为看的是人与人的相似性, 推出来的结果可能更有惊喜

数据更新频率和时效性要求

- 对于item更新时效性较高的产品, 比如新闻, 就无法直接采用item-based的CF, 因为CF是需要批量计算的,在计算结果出来之前新的item是无法被推荐出来的, 导致数据时效性偏低;
- 但是可以采用user-cf, 再记录一个在线的用户item行为对, 就可以根据用户最近类似的用户的行为进行时效性item推荐;
- 对于像影视, 音乐之类的还是可以采用item-cf的;

关于推荐的多样性，有两种度量方法：

- 第一种度量方法是从单个用户的角度度量，就是说给定一个用户，查看系统给出的推荐列表是否多样，也就是要比较推荐列表中的物品之间两两的相似度，不难想到，对这种度量方法，Item CF 的多样性显然不如 User CF 的好，因为 Item CF 的推荐就是和以前看的东西最相似的。
- 第二种度量方法是考虑系统的多样性，也被称为覆盖率(Coverage)，它是指一个推荐系统是否能够提供给所有用户丰富的选择。在这种指标下，Item CF 的多样性要远远好于 User CF, 因为 User CF 总是倾向于推荐热门的，从另一个侧面看，也就是说，Item CF的推荐有很好的新

颖性，很擅长推荐冷门的物品。所以，尽管大多数情况，Item CF 的精度略小于 User CF，但如果考虑多样性，Item CF 却比 User CF 好很多。

用户对推荐算法的适应度

- 对于 User CF，推荐的原则是假设用户会喜欢那些和他有相同喜好的用户喜欢的东西，但如果一个用户没有相同喜好的朋友，那 User CF 的算法的效果就会很差，所以一个用户对的 CF 算法的适应度是和他有多少共同喜好用户成正比的。
- Item CF算法也有一个基本假设，就是用户会喜欢和他以前喜欢的东西相似的东西，那么我们可以计算一个用户喜欢的物品的自相似度。一个用户喜欢物品的自相似度大，就说明他喜欢的东西都是比较相似的，也就是说他比较符合 Item CF 方法的基本假设，那么他对 Item CF 的适应度自然比较好；反之，如果自相似度小，就说明这个用户的喜好习惯并不满足Item CF 方法的基本假设，那么对于这种用户，用 Item CF 方法做出好的推荐的可能性非常低。

基于邻域方法的缺点是：由于实际用户评分的数据是十分稀疏，用户之间可能根本没有相同的评论；而且用启发式的方法很难考虑全面用户和物品之间的所有关系 。

相似用户以及相似物品计算

欧几里得距离：$\text{sim}(x,y)=\frac{1}{1+d(x,y)}$，　$d(x,y)=\sqrt{\left(\sum(x_i-y_i)^2\right.}$

皮尔逊相关系数：

$$p(x,y)=\frac{\sum(x_i-\bar{x})^2(y_i-\bar{y})^2}{\sqrt{\sum(x_i-\bar{x})^2}\sqrt{\sum(y_i-\bar{y})^2}}$$

Cosine相似度：

$$T(x,y)=\frac{x\cdot y}{\|x\|^2\times\|y\|^2}=\frac{\sum x_iy_i}{\sqrt{\sum x_i^2}\sqrt{\sum y_i^2}}$$

Tanimoto系数：

$$T(x,y)=\frac{x\cdot y}{\|x\|^2+\|y\|^2-x\cdot y}=\frac{\sum x_iy_i}{\sqrt{\sum x_i^2}+\sqrt{\sum y_i^2}-\sum x_iy_i}$$