协同过滤在推荐系统中的应用研究

Research on Application of Collaborative Filtering in Recommender Systems

王 霞　(上海金融学院信息管理及计算机应用系　201209)
刘 琴　(上海华东政法学院信息技术中心　200042)

摘要:本文介绍了协同过滤技术,分析了协同过滤在推荐系统中应用时所面临的问题,对主要的解决方法进行了分类研究,最后,对本文研究进行了全面总结。

关键词:协同过滤 推荐系统 算法

1 引言

协同过滤(collaborative filtering)是推荐系统技术中应用最早和最为成功的技术之一。它是基于这样的假设:为一用户找到他真正感兴趣内容的好方法是首先找到与此用户有相似兴趣的其他用户,然后将他们感兴趣的内容推荐给此用户。协同过滤推荐系统就是基于其他用户对某一信息的评价来向用户进行推荐,用户获得推荐是系统从用户购买模式或点击行为等隐式获得的,不需要用户努力地找到适合自己兴趣的推荐信息,如填写一些调查表格等。目前有许多网站采用了基于该技术的推荐系统如:Amazon. com 互联网上最大的书店;CDNow. com Web 上最大的 CD 商店;MovieFinder. com 互联网上最大访问量之一的电影网站等。由微软研究院开发的协同过滤工具已被集成在微软的 Commerce Server 产品中,并被许多站点使用。

2 存在的问题

2.1 数据稀疏性问题

协同过滤技术的实现首先需要使用用户 - 项矩阵(评价矩阵)对用户信息进行表示,尽管这在理论上很简单,但实际上许多电子商务推荐系统要对大量的数据信息进行处理,而在这些系统中一般用户购买商品的总量占网站总商品量的 1% 左右,因此造成了评价矩阵非常稀疏。在这种数据量大而且又稀疏的情况下,一方面难以找到最近邻居用户集,另一方面进行相似性计算的耗费也会很大。

同时,由于数据非常稀疏,在形成目标用户的最近邻居用户集时,往往会造成信息的丢失,从而导致推荐效果的降低。例如:邻居用户关系传递性的丢失,用户 A 与用户 B 相关程度很高,用户 B 与用户 C 相关程度也很高,但由于用户 A 与用户 C 很少对共同的产品进行评价,而认为两者关联程度较低,由于数据的稀疏性,丢失了用户 A 与用户 C 之间潜在的关联。

2.2 算法的可扩展性问题

分析协同过滤算法,全局数值算法能及时利用最新的信息为用户产生相对准确的用户兴趣度预测或进行推荐,但是面对日益增多的用户,数据量的急剧增加,算法的扩展性问题(即适应系统规模不断扩大的问题)成为制约推荐系统实施的重要因素。虽然与基于模型的算法相比,全局数值算法节约了为建立模型而花费的训练时间,但是用于识别"最近邻居"算法的计算量随着用户和项的增加而大大增加,对于上百万的数目,通常的算法会遇到严重的扩展性瓶颈问题。

基于模型的算法虽然可以在一定程度上解决算法的可扩展性问题,但是该类算法往往比较适于用户的兴趣爱好比较稳定的情况,因为它要考虑用户模型的学习过程以及模型的更新过程,对于最新信息的利用比全局数值算法要差些。

协同过滤在推荐系统的实现中,要获得最近邻居用户,必须通过一定的计算获得用户之间的相似度,然后确定最佳的邻居个数,形成邻居用户集。而在这一过程中,如果对全部数据集进行相似性计算,虽然直接,但是运算量和时间花费都极大,无法适应真实的商务系统。如果通过对训练集数据(整个数据集的某一

子集)进行实验获得,虽然不必对整个数据集进行计算,但是必须通过将多次实验结果统计出来才可能得到,这无疑也增加了推荐结果获得的代价和误差。并且如果考虑到数据集的动态变化,这一形成最近邻居用户集技术的实际应用价值越来越小。因此,考虑使用更为有效的最近邻居用户形成办法,对于协同过滤的应用非常必要。

3 解决方法

3.1 LSI/SVD 降维

为了较好地解决协同过滤在推荐系统实现中存在的数据稀疏、同义词(同类产品使用不同的名称进行描述,而无法发现这一相关性)等问题,目前提出了使用在信息检索中被广泛使用的、用于解决同义词和多义词问题的降维技术——隐性语义索引(Latent Semantic Indexing,LSI)。通过降维可以提高数据的密度,发现更多的隐性的用户评价信息。LSI 使用奇异值分解(Singular Value Decomposition,SVD)作为其矩阵分解的算法。SVD 可以很好的与协同过滤技术结合,从而有效的降低数据噪声、发现潜在的关联,而且 SVD 计算可以离线进行。

SVD 可以将一个 m × n 矩阵 R 分解为 3 个矩阵:

$R = T_0 \cdot S_0 \cdot D_0 \text{`} S_0 = \mathrm{diag}(\sigma 1, \cdots, \sigma r)$ (式 3.1)

其中,$\sigma 1 \geq \cdots \geq \sigma r \geq 0$,T0 和 D0 分别是 m × r 和 n × r 的正交矩阵,r 是矩阵 R 的秩($r \leq \min(m,n)$)。S0 是一个 r × r 的对角矩阵,所有的 σr 大于 0 并按照大小顺序排列,称为单值。通常 T0 ,D0, S0 必须是满秩的,将 S0 简化为仅有 k 个单值的矩阵($k < r$)。因为引入了 0,可以将 S0 中的值为 0 的行和列删除,得到一个新的对角矩阵 S,矩阵 T0 ,D0 同样据此简化得到矩阵 T,D,那么有重构的矩阵 RK = TSD′,$RK \approx R$. 单值分解能够生成初始矩阵 R 的所有秩等于 K 的矩阵中与矩阵 R 最近似的一个。

基于维数简化的算法较好的解决了数据稀疏性的问题,同时因为 k < < n,进行预测或者 Top – N 推荐时计算的消耗有相应的降低,也有利于解决扩展性问题,但当数据的量较大时效果并不是很理想。同时,虽然可以离线进行 SVD 的计算,但是训练所需的耗费很大。对于一个 m × n 矩阵(m 为用户数,n 为项数),进行 SVD 计算复杂度为 O((m + n)3),显然用于训练的代价会随着数据量的增大而急剧增加。由于数据动态变化的因素,推荐系统还应考虑重新进行 SVD 计算的频率或者使用增量 SVD 算法(但要考虑使用该算法是否能获得较好的准确度)。从总体上来说,该方法在解决基于用户的协同过滤算法问题时,在结果的准确性上要比运算效率提高上效果明显。

3.2 特征加权(Feature weighting)

使用一些加权的方法来控制不同项(用户信息的描述项或特征项)对用户兴趣度预测的影响,减小甚至消除某些项产生的消极影响,提高与目标项紧密相关项的影响,这样在一定程度上会提高推荐结果的质量。

3.2.1 逆用户频率(Inverse User Frequency)

在信息检索的向量相似性的应用中使用倒排文件频率有效地改善了单纯词频的使用,它的主要思想是减小在文档中经常出现,但对文档主体识别不是非常起作用词语的权重,对出现频率低但对文档主题识别非常有用的词语赋予较高的权重。在[1]中将类似的这种技术运用到协同过滤中并称之为逆用户频率,其主要思想是:对于那些有许多用户评价过的项不如被少数用户评价过的项更有用。由此,倒排用户频率公式如下:

$$\omega_i = \log \frac{n}{n_i} \quad \text{(式 3.2)}$$

n_i 为所有对项 j 进行过评价的用户的总数,n 为数据库中用户的总数,如果所有的用户都对项 j 进行了评价,则 ω_i 的值为 0。当然,如果对所有的项进行评价过的用户数目都相同,那么使用该权重也就没有意义了。

3.2.2 熵(Entropy)

熵用于衡量随机变量的不确定性。在协同过滤算法中用户对某一项或产品评价的分布非常重要,假设如果所有的用户对某产品的评价值都较高,那么计算用户之间的相似性没有意义,因为它说明不了用户之间的区别,但是,如果用户对产品评价在整个范围值内分布分散,用户评价差异较明显,对预测目标用户对该产品的偏爱程度就比较有意义。基于以上的想法,[2]中提出了基于熵的权重方法,公式如下:

$$\omega_i = \frac{H_i}{H_{j,\max}} \text{其中 } H_i = -\sum p_{i,j} \cdot \log_2 p_{i,j} \quad \text{(式 3.3)}$$

公式中 H_i 表示产品 j 的熵,$P_{i,j}$ 表示评价值 i 在对产品 j 的评价中出现的概率,$H_{j,\max}$ 表示假设对产品 j 所

有类型的评价值概率分布相同的情况下的最大熵,使用它是为了减小用户对不同产品进行评价时,由于评价值不同、分散而产生的影响。ω_i 值越大表明用户对产品 i 比较偏爱,该产品对预测的影响较大。然而,如果不同产品间的熵相差不大,同样使用该方法也就没有了意义。例如在电影推荐中人们对每一部电影的爱好程度都会有很大的不同,这样对于许多电影 ω_i 的值可能会为 1,这样基于熵的权重方法失去了作用。

3.2.3 互信息(Mutual Information)

以上两种方法均是从产品或项的自身特点出发来考虑的,并没有涉及到其它项与目标项之间的关系,因为如果项 i 对目标项的预测非常重要,可以赋予它较高的权重,而与目标项相关性程度低的项通过权重降低影响,从而提高推荐系统结果的质量。

文献[2][3]提出使用互信息来衡量不同项之间依赖程度,并以此作为特征权重:

$\omega_i = I(V_i; V_t)$ (式 3.4)

$I(V_i; V_t) = H(V_i) + H(V_t) - H(V_i, V_t)$ (式 3.5)

V_i 与 V_t 分别表示对项 i 与 t 的评价值,$H(V_i, V_t)$ 是两项的联合熵(joint entropy)。由于不是所有的用户均对两项作了评价,因此计算只在对两项均进行了评价的用户中进行。如果在训练数据集中有 n 个这样的用户 m 个项,计算所有项之间的互信息的复杂度为 $O(nm^2)$,而 n 往往远大于 m。

通过在 EachMovie 数据集(1996 ~ 1997 年之间 72916 个用户对 1682 个电影的 2456676 个评价值,其矩阵非常稀疏)上的实验分别对以上三种方法进行了比较,即分别将以上三种权重用于皮而森相关系数的计算中计算相关系数,然后代入预测公式进行计算,所得结果表明在推荐结果的质量上有一定的提高。为了方便,实验任取 10000 个记录而且这些记录中的用户至少对 20 部电影作了评价,将数据集分为训练集(8000 用户)和测试集(2000 用户),实验中使用的评价标准是平均绝对误差 MAE(Mean Absolute Error)来评价预测的准确度,该值越小,表明预测的准确度越高。结果如图 1 所示。

图 1 是在任一用户对某一项评价值已知前提下,利用该用户对其它所有项的评价值来预测该项的值,并进行误差的计算。从实验的数据显示中,我们可以发现使用特征权重 – 互信息对协同过滤算法改进非常明显,大大提高了系统推荐的准确度,其他方法不是很明显,甚至使用倒排用户频率的权重效果还不如原算法好,这主要是因为一方面受数据集的稀疏程度的影响,另外一方面还受相似性计算方法的影响。但随着训练集中用户数量的增多,所有的预测效果都比较稳定,没有出现大幅度的变化,因此,同样可以考虑通过在较小的数据集上的计算获得较为准确的结果。

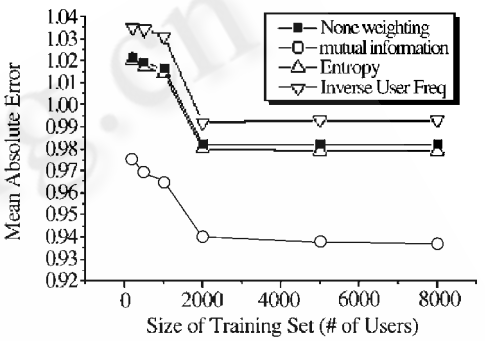

图 1

3.3 用户的筛选

该类方法的主要目的在于解决协同过滤算法的扩展性问题。

基于全局数值的协同过滤算法基于假设是:与目标用户有相似兴趣的用户对目标项的兴趣与目标用户相同。这样,算法成功的关键就在于上述假设是否正确,然而,在实际中这样的假设并不总是成立的。因此,为了提高系统结果的准确度,一方面可以通过给不同项赋予不同的权重,另一方面可以在最近邻居用户的选择上作一定的改进,一方面提高预测的准确度,另一方面减少计算的复杂度,特别是在用户数量较大的情况下。下面介绍两种实现的方法:

(1) 选取具有新颖描述(Novel Profile)的用户。该方法的主要思想是:对于相似性程度差别不大的多个用户,可以只保留其中的一部分,去除的用户对推荐结果的影响可以忽略不计,但由于计算量的减少反而加快了系统运行的速度。算法实现:对于整个训练数据集 T,初始用于预测的用户集中的用户数 Initial_Size,对 T 中的每一项 i,如果所有评价过项 i 的用户集 Ti 中的用户数 > Initial_Size,从 Ti 中随机选 Initial_Size 数目的用户形成初始的用户集 Ti',对于每一个 Ti 中的剩余用户 u,如果 u 的评价值不能通过预测公式在 Ti' 范围内正确的预测到,那么将 u 加入 Ti',这样,对于每一项 i

都有了其相应的缩小了的评价过该项的用户集 Ti'。如果评价值采用从 0 到 5 来进行描述,一般认为如果预测值与实际值的误差范围在 0.5 以内,则认为预测是正确的。Initial_Size 一般设置为 150。

该算法的优点是:

① 充分考虑到了邻居用户的评价值相互之间不一致时,用户评价值变化比较明显的那部分用户;

② 避免了由于多数用户评价值过于集中造成的误差,因为由于数量多这些值往往会比其他最近邻居用户特别是关键的最近邻居用户产生更大的影响,从而导致偏差;

③ 对于新的用户偏好模式能及时根据判断加入到最近邻居用户集中。

实验证明该算法能有效减少每一项进行预测计算的用户数,提高了预测速度和准确度。但是算法也存在不足,由于过多考虑到了评价值比较例外的用户,往往会把一些用户作为最近邻居用户加入,这样的用户对目标项的评价值,即使通过该用户本身对其他项的评价值也无法进行解释,如同数据噪音,导致了预测的失败。另一个不足就是从算法中可以看到由于对 T_i 中的每一用户都要计算在当前最近邻居用户集 T_i' 下的预测值,当数据集很大时,这种耗费会很大。因此该方法往往与其他方法结合,首先进行了一定程度的用户过滤以后,再考虑使用该方法进行进一步的用户过滤。

(2) 选取具有合理描述(Rational Profile)的用户。该算法的主要集中解决的问题是:对任一用户,能否通过他在数据集中的数据较好地描述出来。为方便期间,假设预测用户对项 i 的评价值,$v_{u,i}$ 表示用户 u 对项 i 的评价值,T_i 中用户对项 i 的评价表示为 V_i,用户 u 对其它项的评价值集合表示为 F(u,i) 成为用户 u 的描述项集。T_i 中用户对描述项集中项的评价值集合表示为 $V_{F(u,I)}$。

由 3.2.3 节可知互信息表示项与项的相关性,综合考虑用户描述项与目标项的相关性来进行用户的选择,但实际上并不是用户的描述项越多,用户与目标项之间的合理度(rationality)会越高,因此可使用合理性强度(the strength of rationality of an instance u)来进行用户的选择,公式如下:

$$R_{u,i}^{strength}=\frac{1}{|F(u,i)|}R_{u,i}=\frac{1}{|F(u,i)|}\sum_{i\in F(u,i)}(V_i;V_i)\quad(式\ 3.6)$$

算法实现:首先对项与项之间评价值的互信息进行计算,对每一目标项 i,使用公式(3.6)计算用户集 Ti 中所有用户的合理性强度,按照从高到低排序,然后根据一定的比例 r 选取用户(该比例会影响预测结果的准确性及运行的效率),作为预测时衡量相似性大小选取最近邻居用户的基础,最后利用根据预测公式进行预测计算。

实验证明该方法提高了推荐结果的质量,而且最近邻居用户的数量有明显的减少。在训练阶段(预测计算之前的操作)的代价要比前一种方法低,假设在训练集 T 中有 n 个用户,m 个项,则在训练阶段的计算复杂度为 O(nm2) + O(nm) + O(nlogn),实际上 m 往往比较稳定,随着用户的不断增加 n 的动态变化较大,结果往往会是 n 远远大于 m。而且该算法根据比例 r 值的不同,效率和准确度会有所不同,r 最佳值的获取是这一算法需要考虑的问题。

4 结束语

本文就目前比较典型的对解决协同过滤实现过程中存在问题的方法进行了详细地研究,这些方法各有自己的优缺点,可以针对实际情况选择不同的方法。

参考文献

1 John S. Breese, David Heckerman, Car Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the 14th Conf. on UAI –98, pp. 43 –52, San Francisco, July 24 –26 1998.

2 K. Yu, Z. Wen, X. Xu and M. Ester. Feature Weighting and Instance Selection for Collaborative Filtering. 2nd International Workshop on Management of Information on the Web, in conjunction with the 12th International Conference on DEXA' 2001, Munich, Gerneny, 2001.

3 Kai Yu, Xiaowei Xu, Jianhua Tao, Martin Ester and Hans – Peter Kriegel. Instance selection techniques for memory – based collaborative filtering. In Proceedings of the second international conf. On data mining . part Ⅰ visualization and applications, 2002.

4 蔡登等, 信息协同过滤, 计算机科学, 2002, 29(6): pp. 1 ~ 4。