一种改进相似性度量的协同过滤推荐算法

文俊浩[1,2] 舒 珊[1]
(重庆大学计算机学院 重庆 400044)[1] (重庆大学软件学院 重庆 400044)[2]

摘 要 协同过滤算法是目前电子商务推荐系统中最重要的技术之一,其中相似性度量方法的效果直接决定了推荐系统的准确率。传统的相似性度量方法主要关注用户共同评分项之间的相似度,却忽视了用户共同评分项和用户所有评分项之间的关系。用户共同评分项和用户所有评分项之间的关系可以通过Tanimoto系数来计算,然而Tanimoto系数是基于二值模式下的运算,因此直接运用于推荐系统中的效果并不理想。基于上述问题提出了修正的Tanimoto系数,并将用户共同评分项和用户所有评分项之间的关系融入到传统的相似性度量方法中。实验表明该算法在一定程度上提高了推荐的效率和准确度。

关键词 协同过滤推荐,相似性计算,Tanimoto系数,推荐算法
中图法分类号 TP31 **文献标识码** A

Improved Collaborative Filtering Recommendation Algorithm of Similarity Measure

WEN Jun-hao[1,2] SHU Shan[1]
(College of Computer Science, Chongqing University, Chongqing 400044, China)[1]
(College of Software Engineering, Chongqing University, Chongqing 400044, China)[2]

Abstract Collaborative filtering algorithm is one of the most important technologies in electronic commerce recommendation system. The accuracy of recommendation system directly depends on the effectiveness of the similarity measure. The methods of traditional similarity measure mainly focus on the similarity of user common rating items, but ignore the relationship between the user common rating items and all items the user rates. The relationship between the user common rating items and all items the user rates can be calculated by Tanimoto coefficient. However, Tanimoto coefficient is based on the mode of binary operation, which will not get the satisfactory result if it is directly applied in recommendation system. Aiming at the above problems, the improved Tanimoto coefficient was proposed, and the relationship between the user common rating items and all items the user rates was blended into the traditional similarity measure methods. Experiments show that, to a certain extent, the proposed collaborative filtering algorithm is more effective and accurate.

Keywords Collaborative filtering recommendation, Similarity calculation, Tanimoto coefficient, Recommendation algorithm

1 引言

随着信息技术的快速发展,电子商务也得到了广泛的应用,为了使用户能够在种类繁多的商品中快速准确地找到感兴趣的商品,几乎所有的电子商务平台都不同程度地使用了各种形式的推荐系统。推荐系统的工作,就是收集并分析用户历史的行为信息,根据相关推荐算法对目标用户产生推荐[1]。在众多推荐系统中,协同过滤推荐是迄今为止最为成功也是应用最为广泛的推荐技术[2]。协同过滤的基本思想是:目标用户对有共同爱好的邻居用户所喜欢的商品同样感兴趣,即系统采用相似性度量算法搜索有共同爱好的邻居用户,通过筛选邻居用户的项目信息产生候选推荐项目,并将其推荐给目标用户。传统的协同过滤推荐算法通过计算用户之间的相似度,寻找与目标用户兴趣相似的一组用户作为目标用户的最近邻居,并基于这些最近邻居对目标用户进行推荐,因此,这种算法叫做基于用户(User-Based)的协同过滤算法。Sarwar B等人[3]在此基础上提出了基于项目(Item-Based)的协同过滤算法,该算法从项目的角度进行分析,寻找与目标项目相似的项目集合,然后进行预测和推荐,也取得了不错的效果。

在协同过滤推荐系统中,为了对目标用户产生推荐,需要搜索目标用户最近邻居,因此如何通过评分矩阵来计算项目之间的相似度,是基于项目协同过滤算法的核心部分。最常用的相似度计算方法有余弦相似度[4]、相关相似性[5]以及修

到稿日期:2013-09-17 返修日期:2013-11-21 本文受国家自然科学基金(61075053),教育部高等学校博士学科点科研基金(20120191110028)资助。

文俊浩(1969—),男,博士,教授,主要研究方向为服务计算与面向对象的软件工程;舒 珊(1989—),女,硕士生,主要研究方向为Web服务组合及推荐算法,E-mail:shushan1989@gmail.com。

正的余弦相似度算法。这些相似性度量算法在评分数据丰富的情况下，都能较准确地计算出目标间的相似度。然而近些年随着电子商务系统规模的扩大，用户和商品的数目都急剧增加，真正用户自己评分的项目往往只占了项目总数的1%[6]，导致用于推荐的数据极度稀疏，使用上述相似性度量算法产生的推荐效果将逐步减弱[7]。因此，近年来许多文献致力于提高稀疏数据环境下的推荐效果。主要研究方向分为两部分，一是通过可行方法减少数据稀疏性，二是在稀疏条件不变的情况下提高计算精度。在减少数据稀疏性方面，Xue等人[8]提出评分矩阵补缺算法，运用聚类等方法对用户未评分项目进行初步预测，填补评分矩阵。Choonho等人[9]提出采用多级关联规则挖掘方法解决数据稀疏性问题。虽然这些方法的实验结果表明系统推荐的精度提高了，但是对未评分项填入预测数据无法保证其合理性，这也未从根本上改进相似性度量的问题。在提高计算精度方面，Jamali[10]提出了信任度模型来减少恶意用户的影响，Songjie[11]提出了聚类方法来提升相似性度量精度。这些算法在一定程度上改善了稀疏数据情况下协同过滤推荐的效果，但都有一定的缺陷。本文从充分利用用户已有信息的角度出发，不仅关注了用户之间共同评分项目之间的相似度，还进一步考虑了共同评分项目和用户所有评分项目的关系。通过引入并修正Tanimoto系数，将共同评分项和所有评分项的关系这一特征融合到传统的相似性度量方法中，并取得了不错的效果。

2 传统的相似性度量方法

传统协同过滤算法的基本思想是通过分析评分矩阵，寻找与目标用户(项目)相似的用户(项目)集合，从而产生推荐。用户评分矩阵可以用一个 $m\times n$ 阶的矩阵 $R(m,n)$ 表示，m 行代表 m 个用户，n 列代表 n 个项目。第 i 行第 j 个元素 $R_{i,j}$ 表示用户 i 对项目 j 的评分，用户评分数据矩阵如表1所列。

表1 用户评分矩阵

	$Item_1$	…	$Item_j$	…	$Item_n$
$User_1$	$R_{1,1}$	…		…	
…	…	…		…	…
$User_i$	$R_{i,1}$	…	$R_{i,j}$	…	$R_{i,n}$
…	…	…	…	…	…
$User_m$	$R_{m,1}$	…	$R_{m,j}$	…	$R_{m,n}$

推荐过程主要有3个步骤：建立用户模型，寻找最近邻居，产生推荐。其中寻找最近邻居主要完成目标用户(项目)最近邻居的识别，通过评分矩阵计算用户(项目)间的相似性，然后根据相似性大小找到目标的最近邻居集，最后通过预测公式产生推荐结果，其中准确计算项目间相似度是整个推荐过程的核心。

2.1 传统相似度计算方法

对于目标用户(项目)u，最近邻居识别就是要产生一个以相似度 $\mathrm{sim}(u,v)$ 递减排列的邻居集合 $N_u=\{N_1,N_2,\cdots,N_i\}$，$u\notin N_i$。计算相似度的方法有很多种，其中具有代表性的有以下3种：余弦相似性，相关相似性，修正的余弦相似性。

余弦相似度：将用户 u 对 n 维项目空间的评分当作向量 $\vec{u}$，用户间的相似度取决于评分向量间的余弦夹角，余弦值越大，说明用户间的相似度越高。$\mathrm{sim}(u,v)$ 表示用户 u 与用户 v 的相似度。

$$\mathrm{sim}(u,v)=\cos(\vec{u},\vec{v})=\frac{\vec{u}\cdot\vec{v}}{|\vec{u}||\vec{v}|} \tag{1}$$

相关相似度：相关相似度又称Pearson相关性。设用户 u 与用户 v 共同评分过的项目集合用 R_{uv} 表示，$R_{u,i}$ 表示用户 u 对项目 i 的评分，$\overline{R_u}$ 表示用户 u 对其所有评价过的项目的平均评分值。$\mathrm{sim}(u,v)$ 表示用户 u 与用户 v 的相似度。

$$\mathrm{sim}(u,v)=\frac{\sum\limits_{i\in R_{uv}}(R_{u,i}-\overline{R_u})\times(R_{v,i}-\overline{R_v})}{\sqrt{\sum\limits_{i\in R_{uv}}(R_{u,i}-\overline{R_u})^2}\times\sqrt{\sum\limits_{i\in R_{uv}}(R_{v,i}-\overline{R_v})^2}} \tag{2}$$

修正的余弦相似度：在余弦相似性度量方法中没有考虑到不同用户的评分尺度问题，修正的余弦相似性度量方法通过减去用户对项目的平均评分来改善上述缺陷。设用户 u 与用户 v 共同评分过的项目集合用 R_{uv} 表示，R_u 和 R_v 分别表示用户 u 和用户 v 评过分的项目集合。$\mathrm{sim}(u,v)$ 表示用户 u 与用户 v 的相似度。

$$\mathrm{sim}(u,v)=\frac{\sum\limits_{i\in R_{uv}}(R_{u,i}-\overline{R_u})\times(R_{v,i}-\overline{R_v})}{\sqrt{\sum\limits_{i\in R_u}(R_{u,i}-\overline{R_u})^2}\times\sqrt{\sum\limits_{i\in R_v}(R_{v,i}-\overline{R_v})^2}} \tag{3}$$

2.2 传统相似度计算分析

随着电子商务规模的增长，用户数和项目数呈指数级增长，这导致了用户评分矩阵极度稀疏，用户间共同评分的项目所占比例更小。使用传统的相似度计算很难得到真正的最近邻居集。

首先，只有用户有共同评分项目时，他们的相似度才不为零。在评分矩阵极度稀疏的情况下，余弦相似度计算会将所有未评分的项目填充为缺省值，而真正用户自己评分的项目往往只占了项目总数的1%[6]，因此采用此种方法计算相似度很难获得目标足够大时的最近邻居。

其次，修正的余弦相似度和相关相似度分别考虑的是用户共同评分项的相似度。如果用户的共同评分项很少，例如用户 u 和用户 v 只有一个共同评分项 i 且评分值相同，则使用上述两种相似度计算方法所计算出来的用户 u,v 的相似度很高，这显然是不合理的。

通过以上分析，在用户评分矩阵稀疏的情况下，传统的相似性度量方法并不能有效地度量目标之间的相似性，从而不能准确找到目标的最近邻居，导致推荐系统质量下降。

3 改进的相似性度量方法

传统的相似度计算方法在计算用户间的相似度时只考虑用户间共同项目的评分值，这种方法在用户评分矩阵极度稀疏的情况下很难有效地计算用户间的相似度，从而不能准确地找到目标的最近邻居。为了解决稀疏问题，现有的方法是对用户未评分的项目进行预测评分，通常将其设为一个缺省值，如用户评分的平均分。这种方法虽然对推荐系统的推荐质量有所改善，但是不能从根本上解决数据稀疏的问题，因为用户对未评分项的评分不可能完全相同。因此我们需要从用户已评分项目中获取充分的信息。

在推荐系统中，我们通常只关注目标共同评分的项目，并通过这些共同评分的项目计算目标的相似度，却很少考虑到目标除了共同评分的项目之外已经评分的项目。这样会导致当用户 u 和用户 v 只有一个共同评分项目 i 且评分值相同时，他们的相似度用传统相似性度量的结果会是1。可以做

这样一个假设，如果两个人感兴趣的方向相似，则他们共同评分的项目在两人所有已经评过分的项目中所占的比例应该较大。因此将两者共同评分项目占两者所有评分项目的比例引入到相似性度量中，则可以有效地解决上述问题。本文将共同评分项占所有评分项的比例和传统的相似性度量方法相融合，提出了一种新的相似性度量方法。通过融合共同评分项占所有评分项的比例这一特征，使得新的相似性度量方法在考虑到用户共同评分项目分值之间相似度的同时，考虑了两者整体上感兴趣项目的相似度，有效克服了用户评分数据极度稀疏情况下传统相似性度量方法的不足，使得目标项目的最近邻居更加准确。

下面介绍融合共同评分项占所有评分项比例的协同过滤推荐算法。算法主要分为两步：根据相似度寻找最近邻居集和根据最近邻居集产生推荐结果。

3.1 寻找最近邻居集

通过计算目标间的相似度来寻找目标的最近邻居集。对于一个具体的目标产生一个依据相似度大小降序排列的集合：$N_u=\{N_1,N_2,\cdots,N_i\}$，$u\notin N_i$，其中 N_u 表示项目 u 的 i 个邻居集合。传统的相似度计算方法如 2.1 节介绍的几种，针对 2.2 节中分析的传统相似度度量方法在数据稀疏情况下的不足，本文引入共同评分项目占所有评分项目比例这一特征，并将其与传统的相似度计算方法结合，以提高相似性度量和预测评分的准确性。

3.1.1 共同评分项目占用比

针对共同评分项目占用比可以从整体上衡量两个目标的相似度，我们将共同评分项目占用比引入相似性度量。从用户对项目评分的角度来看，共同评分项目占用比越高，则说明两个人感兴趣的方向越相似；共同评分项目占用比越低，则说明两个人感兴趣的方向越不同。因此，目标 u,v 之间的共同评分项目占用比 $T(u,v)$ 可以用 Tanimoto 系数表示：

$$T(u,v)=\frac{\sum_{i=1}^{N}(u_i\wedge v_i)}{\sum_{i=1}^{N}(u_i\vee v_i)} \tag{4}$$

其中，u_i 和 v_i 表示用户 u 和 v 是否对项目 i 进行评分，如评分则其值为 1，若未评分则其值为 0。$\wedge$ 和 $\vee$ 则是位运算中的与和或运算。T 越大则 u 和 v 的整体相似度越高，T 越小则 u 和 v 的整体相似度越低。

然而 Tanimoto 系数是一种二值模式下的运算，仅仅考虑了用户是否对项目进行了评分，却未考虑用户对项目评分的差异。在推荐系统中，用户对项目的评分范围往往很广，若用户评分矩阵如表 2 所列，则使用 Tanimoto 系数计算出来的用户 A 和用户 B 之间的相似度，与用户 C 与用户 B 之间的相似度完全相同。因此本文提出了修正的Tanimoto系数，使其可以扩展到非二值模式下进行评分。

表 2 用户评分矩阵

	Ⅰ	Ⅱ	Ⅲ	Ⅳ	Ⅴ
用户 A	5	0	4	3	2
用户 B	5	5	4	3	2
用户 C	1	0	2	3	4

由于推荐系统中的评分分值往往分布在 1 到 N 之间，因此我们不能直接使用 Tanimoto 系数计算相似度。对此本文通过考虑用户共同评分项之间的差异提出了修正的 Tanimoto 系数，计算方法如下：

$$T(u,v)=\frac{\sum_{i\in R}\left(1-\sqrt{\left(\frac{u_i-v_i}{N}\right)^2}\right)}{\sum_{i=1}^{m}(u_i\wedge v_i)} \tag{5}$$

其中，R 表示用户 u 和用户 v 的共同评分项集合，u_i 和 v_i 表示用户 u 和 v 对项目 i 的评分，m 表示评分集合的数量，N 表示系统评分所允许的最大分值。在此计算方法中，我们考虑了用户对同一项目的评分差异，只有评分完全相同时，用户对此项目的相似度才为 1。

3.1.2 融入修正的 Tanimoto 系数的相似性度量

传统的相似性计算方法是基于目标间共同评分项目的，在评分数据非常丰富的情况下，共同评分项目很多，传统的相似性计算方法能够准确地计算出目标间相似度；而在数据稀疏的情况下，目标已评分项目占所有项目的比例就很小，而目标之间共同评分项目则更少，传统的相似性度量方法就很难准确衡量项目间的相似性。本文提出的融合修正的 Tanimoto 系数的相似性度量方法考虑了共同评分项目占所有评分项目的比例，因此可以有效地缓解数据稀疏的情况下用传统相似性度量方法计算目标间相似性不够准确的问题。

将修正的 Tanimoto 系数与传统相似性度量通过公式组合，得到的就是融合修正 Tanimoto 系数的相似度：

$$\mathrm{sim}_T(u,v)=\alpha\times T(u,v)+(1-\alpha)\times\mathrm{sim}(u,v) \tag{6}$$

其中，$T(u,v)$ 就是目标 u 和 v 之间的修正的 Tanimoto 系数，$\mathrm{sim}(u,v)$ 则是 3.1 节中提到的传统的相似性度量方法。参数 α 为松弛因子，用来调节修正的 Tanimoto 系数在相似性度量时所占的比重，在实验部分会分析 α 的取值对相似性度量的影响。$\mathrm{sim}_T(u,v)$ 就是融合了修正的 Tanimoto 系数以及传统相似性度量的方法计算出来的目标 u 和 v 之间的相似性。融合了修正的 Tanimoto 系数以及传统相似性度量的方法考虑了共同评分项分值相似度的同时考虑了共同评分项在所有评分项中所占的比例，这在一定程度上缓解了数据稀疏问题，有效地遏制了在共同评分用户极少的情况下，直接用传统相似度度量方法计算项目间相似度不够准确的问题。

3.2 产生推荐结果

利用本文提出的融合 Tanimoto 系数的相似度计算方法计算出项目间相似度值从而得到目标项目的最近邻居，下一步需要预测目标用户对项目的评分，最后根据评分大小产生推荐。设项目 i 的最近邻居集为 $N(i)$，则用户 u 对项目 i 的预测评分 $P_{u,i}$ 可以根据目标用户 u 对最近邻居集 $N(i)$ 中的项目评分得到，计算方法如下：

$$P_{u,i}=\overline{R}_i+\frac{\sum_{j\in N(i)}\mathrm{sim}(i,j)\times(R_{u,j}-\overline{R}_j)}{\sum_{j\in N(i)}|\mathrm{sim}(i,j)|} \tag{7}$$

其中，$\mathrm{sim}(i,j)$ 表示项目 i 与其最近邻居 j 之间的相似度，$\overline{R}_i$ 和 $\overline{R}_j$ 分别表示用户对项目 i 和项目 j 的评分。

4 实验结果及分析

4.1 数据集

实验数据采用开源的美国 Minnesota 大学 GroupLens 项目组提供的 Movielens 数据集，该数据集已经被广泛应用在推荐系统的测评中。其包含了 943 名用户对 1682 部电影的

超过100000的评价，每个用户至少评价过20部电影，分值为1到5的整数，分数越高表明用户对电影的评价越高，少于20个评价的用户从数据集中被清除。该数据集由3张数据表组成：User(用户表)、Item(电影表)、Rate(用户评分表)。实验数据分析如表3所列。

表3　实验数据分析

用户总数	943
电影总数	1682
评分总数	100000
用户最大评分项	752
用户最小评分项	20
用户平均评分项	106.04
稀疏等级	0.937

整个数据集被划分为训练集和测试集，训练集数据作为评分矩阵，用于算法输入并进行评分预测，而测试集数据用于测试算法性能。

4.2　衡量标准

本文采用通常使用的平均绝对偏差(Mean Absolute Error，MAE)作为准确性度量标准。MAE通过计算预测的用户评分与实际的用户评分之间的偏差，即用户对资源评分实际值与预测值的绝对值的加权平均值，来度量预测的准确性，MAE越小，推荐结果的准确度越高。MAE计算公式如下：

$$MAE=\frac{\sum_{i=1}^{n}|P_i-R_i|}{n} \tag{8}$$

其中，算法预测出的用户评分集合为$\{P_1,P_2,\cdots,P_n\}$，而对应测试集中实际用户评分集合为$\{R_1,R_2,\cdots,R_n\}$，n表示集合大小。

4.3　实验方案

1)调整参数α取得的MAE

由于本文提出的相似性度量中的α为可调节参数，因此在α取不同值时，相似性度量所获得的效果也不同。在实验中，α取值范围为[0,1]，每次增加0.25，最近邻居数目取值范围为[5,50]，观察MAE的变化。实验结果如图1所示。从图1中可看出，α取值为0.25时效果最好。

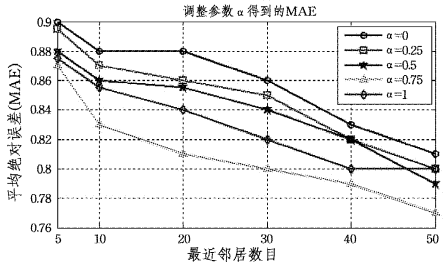

图1　α因子对MAE的影响

2)不同相似性度量算法下的MAE

本次实验将本文提出的相似性度量方法与传统相似性度量方法计算出的MAE进行了比较，实验中α因子取值为0.25，最近邻居数目取值范围为[5,50]。实验结果如图2所示。

从实验结果中可以看出，本文提出的融合了修正的Tanimoto系统的相似性度量方法所计算出的MAE值较小，确实在推荐精度上有一定的改进。

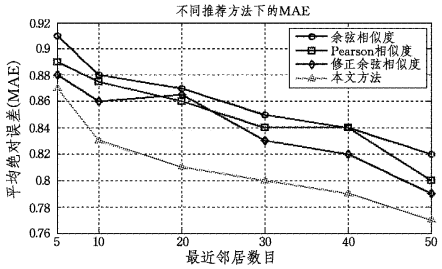

图2　不同推荐方法下的MAE

结束语　协同过滤推荐系统通过计算用户评分项之间的相似度来产生推荐结果。本文通过考虑用户共同评分项和共同评分项在所有评分项中所占比例两个因素，提出了一种新的相似性度量算法。该算法有效地解决了传统相似性度量算法由于共同评分项少而引起的推荐结果不准确这一问题，同时在一定程度上提升了推荐质量。

参 考 文 献

[1] Varian J, Resnick P. Recommendation systems [J]. Mineapplio: Communications of the ACM, 1997, 40(3): 56-58

[2] Adomavicius G, Tuzhilin A. Towards the next generation of recommender system: a survey of the state-of-the-art and possible extensions [J]. IEEE Transaction on Konwledge and Data Engineering, 2005, 17(6): 734-749

[3] Sarwar B, Karypis G, Konstan J A. Item-based collaborative filtering recommendation algorithms [C]// Proceedings of the 10th International World Wide Web Conference (WWW10). Hong Kong, 2001: 285-295

[4] Salton G, McGill M. Introduction to modern information retrieval [M]. New York, USA: McGraw-Hill, 1983

[5] Resnick P, Iacovou N, Suchak M. An open architecture for collaborative filtering of net news [C]// Proc. of ACM Conference on Computer Supported Cooperative Work. 1994

[6] Resnick P, Iacovou N, Suchak M. Grouplens: an open architecture for collaborative filtering of netnews [C]// Proceedings of ACM CSCW 94 Conference on Computer-Supported Cooperative Work. 1994: 175-186

[7] 邓爱林，朱扬勇，施伯乐. 基于项目评分预测的协同过滤推荐算法[J]. 软件学报，2003，14(9)：1621-1628

[8] Xue G R, Lin C, Yang Q. Scalable collaborative filter using cluster-based smoothing [C]// Proc. of SIGIR. 2005

[9] Choonho K, Juntae K. A recommendation algorithm using multi-level association rules [C]// Proceedings of the IEEE/WIC International Conference on Web Intelligence (WI03). 2003

[10] Jamali M, Ester M. Trustwalker: a random walk model for combining trust-based and item-based recommendation [C]// KDD 2009. 2009

[11] Songjie G, Chongben H. Employing fuzzy clustering to alleviate the sparsity issue in collaborative filtering recommendation algorithm [C]// Proceeding of 2008 International Pre-Olympic Congress on Computer Science. World Academic Press, 2008: 449-454