基于用户近邻约束的矩阵因子分解算法

查 九,李振博,徐桂琼
(上海大学 管理学院,上海 200444)

摘 要:矩阵因子分解推荐算法是基于模型的协同过滤算法中应用最广泛的一种推荐技术。针对推荐系统数据的稀疏性和推荐算法的实时性等问题,在传统矩阵因子分解模型的基础上引入用户近邻模型约束,提出基于用户近邻约束的矩阵因子算法。该算法充分利用了矩阵因子模型的优点,通过用户近邻约束进一步提高了算法相应的实时性和推荐的质量。在 MovieLens 数据集上的实验结果表明,该算法能有效解决数据稀疏和实时性问题,在推荐质量上比传统算法有了较大提高。

关键词:协同过滤;用户近邻;近邻约束;矩阵因子
中图分类号:TP391 **文献标识码**:A **文章编号**:1673-629X(2015)06-0001-05
doi:10.3969/j.issn.1673-629X.2015.06.001

A Matrix Factorization Algorithm Based on User's Neighbors Regularized

ZHA Jiu, LI Zhen-bo, XU Gui-qiong
(School of Management, Shanghai University, Shanghai 200444, China)

Abstract: Matrix factorization algorithm based on collaborative filtering is one of the most widely used in the personalized recommendation system. Concerning the problems of data sparsity and real-time in recommendation system, a matrix factorization algorithm based on user's neighbors regularized is proposed based on traditional matrix factorization model. The algorithm takes advantage of the matrix factorization model, using the user's neighbor as a regularization to improve the quality and real-time of recommendation algorithm. The experimental results in movieLens datasets show that the proposed algorithm can more efficiently improve recommendation quality than the traditional algorithm, and solve the problems of data sparsity and real-time.

Key words: collaborative filtering; user's neighbors; neighbor regularization; matrix factorization

0 引 言

随着互联网全面普及和电子商务的快速发展,信息过载现象越来越严重,大量的商品信息增加了客户购买所需物品的难度,消费者很难快速且有效地做出决策。为了让顾客在购买产品时尽可能少地浏览无关信息,解决信息过载问题,个性化推荐技术应运而生[1]。

基于内存的协同过滤推荐算法[2]是个性化推荐技术中研究和使用最为广泛的一种推荐算法。其核心思想是基于目标用户的邻居的资料来获得目标用户的推荐,首先在用户群中找到指定用户的邻居(兴趣)用户,然后根据邻居用户的评分预测目标用户对商品项的评分值,选择评分项最高的前 N 项商品反馈给目标用户[3-6]。基于内存的协同过滤算法也有很多局限,例如当数据集十分稀疏时,基于用户之间共同评分的项目很少,计算出来的相似度不能准确地反映用户之间的关系;随着用户和项目数量的扩大,将花费更多的计算资源在最近邻搜索上,难以保证系统的实时性,无法发现商品之间存在的隐含关系。

为克服基于内存协同过滤算法的局限性,使得算法具有更好的可扩展性,研究人员后来提出了基于模型的协同过滤算法(Model-based CF)。基于模型的方法有聚类模型方法[7]、矩阵分解[8]和图论模型[9]等。在基于模型的协同过滤算法中,矩阵因子分解(Matrix

收稿日期:2014-07-17 **修回日期**:2014-10-23 **网络出版时间**:2015-05-06
基金项目:国家自然科学基金资助项目(11201290)
作者简介:查 九(1990-),男,硕士研究生,研究方向为数据挖掘、个性化推荐;徐桂琼,副教授,研究方向为复杂系统建模与动力学研究、数据挖掘、协同过滤。
网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20150506.1630.023.html

Factorization,MF)算法以其稳定性和准确性吸引了更多的研究关注[10-11]。矩阵因子方法能解决数据稀疏的问题,但是如何从稀疏矩阵中获取有用信息进一步取得更好的推荐效果对于矩阵因子方法来说至关重要。文中在传统的矩阵因子算法基础上,将用户的近邻作为矩阵因子模型的约束得到基于近邻的矩阵因子分解模型,通过该模型来达到准确预测用户偏好的目的。在MovieLens公开数据集上的实验结果表明,文中提出的基于近邻的矩阵因子分解算法可以获得更好的推荐效果。

1 相关算法概述

1.1 基于用户最近邻协同过滤算法概述

在一个基于用户的协同过滤算法[12]中,输入数据为用户评分矩阵 $\boldsymbol{R}$($\boldsymbol{R}$ 表示 m 个用户对 n 个项目评分组成的稀疏矩阵)。基于用户协同过滤算法的第一步是根据评分矩阵 $\boldsymbol{R}$ 计算用户之间的相似度 $\mathrm{sim}(i,v)$。通常,计算两个用户之间的相似度主要有3种方法[13]:余弦相似性、修正余弦相似性、Pearson相关相似性。

余弦相似性(Cosine Correlation,CC):用户对 n 个项目的评分可视为 n 维向量,用户 i 和用户 v 的相似性即为相应两个 n 维向量的夹角余弦。

R_{ij}、R_{vj} 分别表示用户 i、v 对 j 的评分,用户 i、v 在项目集 I 上共同评分的项目集为 $I_{iv}=\{j\in I \mid R_{ij}\neq 0 \cap R_{vi}\neq 0\}$。

$$\mathrm{sim}(i,v)=\frac{\sum_{j\in I_{iv}}R_{ij}R_{vj}}{\sqrt{\sum_{j\in I_{iv}}R_{ij}^2}\sqrt{\sum_{j\in I_{iv}}R_{vj}^2}} \tag{1}$$

修正余弦相似性(Adjusted Cosine Correlation,ACC):考虑了用户的评分尺度问题,可由如下公式定义,其中 $\bar{R}_i$ 表示用户 i 的平均评分,$\bar{R}_v$ 表示用户 v 的平均评分。

$$\mathrm{sim}(i,v)=\frac{\sum_{j\in I_{iv}}(R_{ij}-\bar{R}_i)(R_{vj}-\bar{R}_v)}{\sqrt{\sum_{j\in I_i}(R_{ij}-\bar{R}_i)^2}\sqrt{\sum_{j\in I_v}(R_{vj}-\bar{R}_v)^2}} \tag{2}$$

Pearson相关相似性(Pearson Correlation,PC):通过计算两个用户的Pearson相关系数来确定它们之间的相似性,Pearson相关相似性可由如下公式计算得到。

$$\mathrm{sim}(u,v)=\frac{\sum_{i\in I_{uv}}(R_{ui}-\bar{R}_u)(R_{vi}-\bar{R}_v)}{\sqrt{\sum_{i\in I_{uv}}(R_{ui}-\bar{R}_u)^2}\sqrt{\sum_{i\in I_{uv}}(R_{vi}-\bar{R}_v)^2}} \tag{3}$$

基于用户协同过滤算法的第二步是根据计算得到的用户相似度,找到当前用户的最近邻居并产生预测值。对目标用户 i 而言,在整个评分矩阵空间中搜索出 k 个相似度最高的用户即可以组成其最近邻居集合 $N(i)$。

最后根据当前用户 k 个最近邻居对项目的评分信息预测当前用户对其未评分项目的评分值,以此产生Top-N 推荐。用户 i 对未评分项目 j 的预测评分 p_{ij} 可以通过用户 i 的最近邻居集合 $N(i)$ 对 j 的评分得到,计算方法如下:

$$p_{ij}=\bar{R}_i+\frac{\sum_{v\in N(i)}\mathrm{sim}(i,v)\cdot(R_{vj}-\bar{R}_v)}{\sum_{v\in N(i)}|\mathrm{sim}(i,v)|} \tag{4}$$

1.2 矩阵因子分解算法概述

矩阵因子分解算法(Matrix Factorization)是一种十分有效的协同过滤推算法[14-17],G Takács在文献[18]中提到的几种基本的矩阵因子分解算法在大的数据集上效果更明显。基于矩阵因子分解的协同过滤算法是将用户-项目评分矩阵 $\boldsymbol{R}(m,n)$ 分解成 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的乘积形式:

$$\boldsymbol{R}\approx\boldsymbol{U}^{\mathrm{T}}\boldsymbol{V} \tag{5}$$

其中,$\boldsymbol{U}\in R^{l\times m}$,$\boldsymbol{V}\in R^{l\times n}$,$l<\min(m,n)$。

传统上矩阵分解的SVD方法[19]是采用公式(6)最小化逼近评分矩阵 $\boldsymbol{R}$:

$$\frac{1}{2}\|\boldsymbol{R}-\boldsymbol{U}^{\mathrm{T}}\boldsymbol{V}\|_{\mathrm{F}}^2 \tag{6}$$

其中,$\|\cdot\|$ 表示尼乌斯范式,只需分解稀疏评分矩阵 $\boldsymbol{R}$,将公式转化成如下形式:

$$\min_{U,V}\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}I_{ij}(R_{ij}-\boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j)^2 \tag{7}$$

其中,R_{ij} 为用户实际对项目的评分,$R_{ij}\neq 0$ 时 $I_{ij}=1$,$R_{ij}=0$ 时 $I_{ij}=0$。为了防止过拟合,还要加上正则化项 $\frac{1}{2}(\lambda_1\|\boldsymbol{U}\|_{\mathrm{F}}^2+\lambda_2\|\boldsymbol{V}\|_{\mathrm{F}}^2)$ 以达到 $\boldsymbol{R}\approx\boldsymbol{U}^{\mathrm{T}}\boldsymbol{V}$ 的目的[20],最终的优化目标损失函数为:

$$L(\boldsymbol{U},\boldsymbol{V})=\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}I_{ij}(R_{ij}-\boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j)^2+\frac{1}{2}(\lambda_1\|\boldsymbol{U}\|_{\mathrm{F}}^2+\lambda_2\|\boldsymbol{V}\|_{\mathrm{F}}^2) \tag{8}$$

其中,λ_1、λ_2 为惩罚因子,是为了避免数据量不足和训练过度拟和的现象,通过最小化损失函数 $L(\boldsymbol{U},\boldsymbol{V})$ 求出 $\boldsymbol{U}$、$\boldsymbol{V}$ 矩阵的最优解。

2 基于用户近邻约束的矩阵因子分解算法

第1节中基于用户最近邻的协同过滤算法:通过找到目标用户的最近邻,利用最近邻加权平均来预测目标用户对项目的评分。基于用户的协同过滤算法认

为目标用户的近邻用户加权平均可以代替目标用户。杨阳等在文献[21]中提出一种矩阵分解和用户最近邻结合的模型,通过该模型预测用户的评分值为:

$$R_{ij}^{p} = \boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j + \sum_{v\in N}\mathrm{sim}(i,v)(R_{ij} - R_{iv})$$

其中,N 是用户 i 的近邻模型。该模型虽部分解决了预测准确度的问题,但需要对每个用户计算最近邻,算法的时间复杂度在原有的基础上扩大了 k 倍,为 $O(m*n*k)$(k 是用户的近邻数目),使得在推荐系统中实时性得不到很好的响应,影响用户体验。

为解决该模型存在的问题,文中提出基于用户近邻约束的矩阵因子分解算法(Matrix Factorization Based on User's Neighbors Regularized,UB_MF)。该算法中将用户近邻模型作为约束因子加入到矩阵因子分解模型中。

矩阵因子分解模型中,分解得到用户特征矩阵 $\boldsymbol{U}$ 和项目特征矩阵 $\boldsymbol{V}$。$\boldsymbol{U}_i$ 为用户 i 在 l 个隐性因子上的特征向量,用户 i 的近邻向量为 $\boldsymbol{U}_v$,其中 $v \in N(i)$,$N(i)$ 为用户 i 的近邻用户。结合第 1 节介绍的基于用户的协同过滤算法,文中将用户近邻作为约束加入到矩阵因子分解模型中,提出基于用户近邻约束的矩阵因子分解算法。该模型如下所示,$\tilde{L}(\boldsymbol{U},\boldsymbol{V})$ 是损失函数,$\tilde{e}$ 为用户近邻约束:

$$\tilde{L}(\boldsymbol{U},\boldsymbol{V}) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - \boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j)^2 + \frac{1}{2}(\lambda_1 \|\boldsymbol{U}\|_{\mathrm{F}}^2 + \lambda_2 \|\boldsymbol{V}\|_{\mathrm{F}}^2) + \tilde{e} \quad (9)$$

其中,$\tilde{e} = \frac{\beta}{2}\sum_{i=1}^{m}\left\|\boldsymbol{U}_i - \frac{\sum_{v\in N(i)}\mathrm{sim}(i,v)\boldsymbol{U}_v}{\sum_{v\in N(i)}|\mathrm{sim}(i,v)|}\right\|_{\mathrm{F}}^2$;$\beta(\beta>0)$ 表示用户近邻对目标函数的影响程度,β 越大表示用户近邻在推荐算法中占有更重要的角色,β 越小表示所起到的作用越小。

因为 $|\mathrm{sim}(i,v)|$ 是用户 i 和其近邻用户 v 在原始评分矩阵 $\boldsymbol{R}$ 上计算的相似度,在用户特征矩阵 $\boldsymbol{U}(m*l)$ 中,并不能直接反映用户 i 的特征向量 $\boldsymbol{U}_i$ 与其近邻 v 的特征向量 $\boldsymbol{U}_v$ 的关系。如果 $\boldsymbol{U}_i$ 与 $\boldsymbol{U}_v$ 有较大差异,将会导致信息丢失问题,为此将 $\tilde{e}$ 进行改进:

$$\tilde{e} = \frac{\beta}{2}\sum_{i=1}^{m}\sum_{v\in N(i)}\mathrm{sim}(i,v)\|\boldsymbol{U}_i - \boldsymbol{U}_v\|_{\mathrm{F}}^2 \quad (10)$$

最终的损失函数为:

$$\tilde{L}(\boldsymbol{U},\boldsymbol{V}) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - \boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j)^2 + \frac{\beta}{2}\sum_{i=1}^{m}\sum_{v\in N(i)}\mathrm{sim}(i,v)\|\boldsymbol{U}_i - \boldsymbol{U}_v\|_{\mathrm{F}}^2 + \frac{1}{2}(\lambda_1 \|\boldsymbol{U}\|_{\mathrm{F}}^2 + \lambda_2 \|\boldsymbol{V}\|_{\mathrm{F}}^2) \quad (11)$$

该模型算法时间复杂度为 $O(m*n)$,算法的效率得到了有效提高。为求得 $\boldsymbol{U}$、$\boldsymbol{V}$ 矩阵的最优解,需最小化损失函数 $\tilde{L}(\boldsymbol{U},\boldsymbol{V})$。利用梯度下降法来达到最小化目的,该算法步骤如下所示:

(1)对 $\tilde{L}(\boldsymbol{U},\boldsymbol{V})$ 分别求偏导得到梯度。

$$\frac{\partial}{\partial(\boldsymbol{U}_i)}\tilde{L} = \sum_{j=1}^{n} I_{ij}(R_{ij} - \boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j)(-\boldsymbol{V}_j) + \beta\sum_{v\in N(i)}\mathrm{sim}(i,v)(\boldsymbol{U}_i - \boldsymbol{U}_v) + \lambda_1\boldsymbol{U}_i$$

$$\frac{\partial}{\partial(\boldsymbol{V}_j)}\tilde{L} = \sum_{i=1}^{m} I_{ij}(R_{ij} - \boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j)(-\boldsymbol{U}_i) + \lambda_2\boldsymbol{U}_j \quad (12)$$

设置参数 θ 为相似度的阈值,$\mathrm{sim}(i,v)$ 的取值为 $\mathrm{sim}(i,v) \geqslant \theta$,表示只计算和用户 i 相似度值大于 θ 的近邻。例如 $\theta = 0.5$ 时,表示只取和目标用户 i 相似度大于等于 0.5 的近邻用户。

(2)更新 $\boldsymbol{U}$、$\boldsymbol{V}$。

$$\boldsymbol{U}_i = \boldsymbol{U}_i - \alpha\frac{\partial}{\partial(\boldsymbol{U}_i)}\tilde{L}$$

$$\boldsymbol{V}_j = \boldsymbol{V}_j - \alpha\frac{\partial}{\partial(\boldsymbol{V}_j)}\tilde{L} \quad (13)$$

其中,α 为梯度下降算法中的学习速率,它的取值一般为 0.001 左右。

(3)经过一轮迭代后((1)、(2)两步),损失函数 $\tilde{L}(\boldsymbol{U},\boldsymbol{V})$ 的值达到要求或者经过设定的迭代轮次则迭代终止,否则重复(1)、(2)两步。

在样本集上运用上述算法进行训练,迭代终止后得到最终的矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$。$\boldsymbol{U}$ 代表用户在 l 个隐性因子上面的特征,$\boldsymbol{V}$ 代表项目在 l 个隐性因子上面的特征,预测目标用户 i 对项目 j 的分值为 $R_{ij}^{p} = \boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j$。通过矩阵因子分解方法有效降低了时间和空间的复杂度,再结合用户历史评分基础上选取的近邻模型,使得用户之间预测的相关性进一步提高,从而有效地提高了预测的准确性。

3　实验分析

3.1　实验数据集和度量标准

实验以常用的 MovieLens 数据集作为测试数据,由美国 Minnesota 大学的 GroupLens 研究小组提供 ML100k 数据集。该数据集包含了 943 个用户对 1 682 部电影在连续 7 个月内的评分数据,其中每个用户至少对 20 部电影进行了评分。实验需要将整个数据集划分为训练集和测试集,对 ML100k 提供的 10 万条评分数据,随机选取其中 80% 用作训练集,另外 20% 用作测试集。数据集的稀疏等级(未知评分在整个数据

集所占的比例)为:

1-100 000/943×1 682=93.7%

评分预测的准确度一般通过均方根误差(RMSE)和平均绝对误差(MAE)计算。均方根误差和平均绝对偏差越小,预测的准确度越高。对于测试集中的一个用户 i 和物品 j,令 r_{ij} 是用户 i 对物品 j 的实际评分,而 $p_{ij}=\boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j$ 是迭代得到的矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 计算给出的预测评分值,那么 RMSE 的定义为:

$$\mathrm{RMSE}=\sqrt{\frac{\sum_{i,j\in T}(r_{ij}-p_{ij})^2}{|T|}} \tag{14}$$

MAE 采用绝对值计算预测误差,它的定义为:

$$\mathrm{MAE}=\frac{\sum_{i,j\in T}|r_{ij}-p_{ij}|}{|T|} \tag{15}$$

3.2　实验结果与分析

本节通过实验来检验文中推荐算法的推荐质量;讨论维度 l、约束因子参数 β、不同相似度计算方式和相似度阈值对推荐结果的影响,并和传统矩阵因子分解算法进行比较。

(1)迭代次数对传统矩阵因子分解算法的影响。

文中采用梯度下降算法训练出矩阵 $\boldsymbol{U}$、$\boldsymbol{V}$ 需要确定的迭代次数。设置算法参数:学习速率 $\alpha=0.001$,惩罚因子 $\lambda=\lambda_1=\lambda_2=0.01$,约束因子 $\beta=0.01$。在 ML100k 训练集上,分别选取维度为 $k=10$,$k=13$,$k=15$,在不同迭代次数 f 下计算预测评分矩阵 $\tilde{\boldsymbol{R}}$ 和原始评分矩阵 $\boldsymbol{R}$ 的 RMSE。RMSE 随迭代次数 f 变化如图1所示。

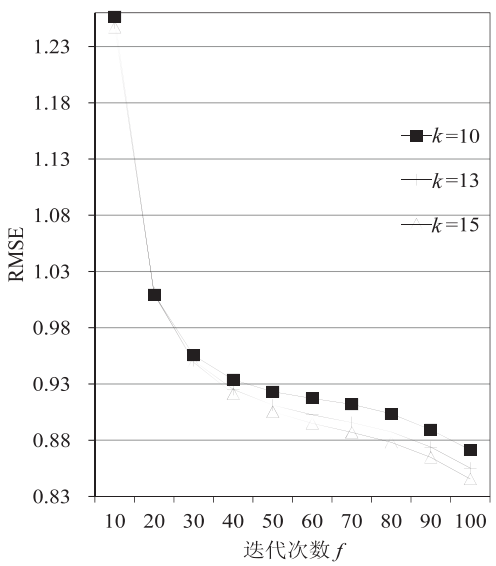

图1　RMSE 随迭代次数 f 的变化曲线

通过图1发现,随着迭代次数的增加,RMSE 越来越小,也趋于稳定状态,说明迭代得到的 $\boldsymbol{U}$、$\boldsymbol{V}$ 矩阵乘积 $\boldsymbol{U}^{\mathrm{T}}\boldsymbol{V}$ 和原始矩阵 $\boldsymbol{R}$ 越来越相近了。迭代次数增加能有效提高预测的准确率,但迭代次数增加得到矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的时间也随之增加。

(2)基于用户近邻约束的矩阵因子分解算法。

为比较基于用户近邻约束的矩阵因子分解(UB_MF)和传统矩阵因子分解(MF)算法的推荐效果,分别利用两种算法在 ML100k 训练集上训练得到 $\boldsymbol{U}$、$\boldsymbol{V}$ 矩阵,在 ML100k 测试集上计算平均绝对误差(MAE)来检验算法的优劣。

在 UB_MF 算法中,分别选取用户 CC、PC 和 ACC 作为用户的相似度计算方法。对于两种算法选择共同参数:学习速率 $\alpha=0.001$,惩罚因子 $\lambda=\lambda_1=\lambda_2=0.01$,约束因子 $\beta=0.01$,迭代次数 $f=100$ 进行实验。在 UB_MF 算法中,为了取得较好的效果,统一取相似度阈值 $\theta=0.5$,分别在不同维度 $l(l=9,10,\cdots,15)$ 上进行实验。得到的 MAE 值随着维度 l 变化如图2所示。

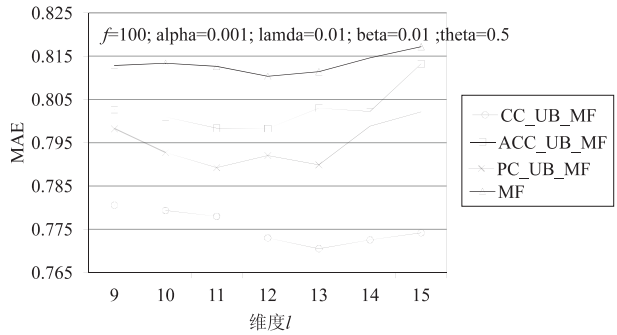

图2　$f=100$ 下 MAE 随维度 k 变化曲线

在图2中,选取特定迭代次数 $f=100$ 与不同相似度计算出来的结果。在 UB_MF 算法中,不管选取哪种相似度,最终的预测结果计算出来的 MAE 都比 MF 效果要好。由此可知与传统的 MF 算法相比,文中提出的 UB_MF 可以较好地提高推荐系统的推荐质量。在 UB_MF 算法中,选择合适的用户相似度计算方法,也能有效地提高推荐质量。该实验中,选择 Cosine 余弦相似度计算方法的 CC_UB_MF 算法的推荐效果相对其他相似度计算方法的推荐效果较好。

(3)不同相似度阈值 θ 对算法的影响。

在(2)中选择的相似度阈值 $\theta=0.5$,就是说在选择用户的近邻时,只选择两者相似度值大于0.5的近邻用户。固定维度 $l=12$,通过改变相似度阈值,分别取 θ 为0.1,0.2,…,0.9,其余参数和(2)中实验保持一致。在3种相似度计算方法下,得到的基于用户近邻约束的矩阵因子分解算法 MAE 随着阈值 θ 变化曲线如图3所示。

通过实验(3)发现,相似度阈值 θ 的取值会对文中提出的算法有影响。文中提出的算法在3种相似度计算方法下得到的 MAE 值都随着 θ 从0.1~0.9变化,先逐渐下降,后逐渐上升。θ 是用户之间的相似度阈值,如果取值较低,则用户近邻数目会增加,可能出现过度拟合情况;如果取值过高,用户近邻也会减少,

用户的近邻对该目标用户起到的约束作用就不是很大,导致出现预测精度的问题。

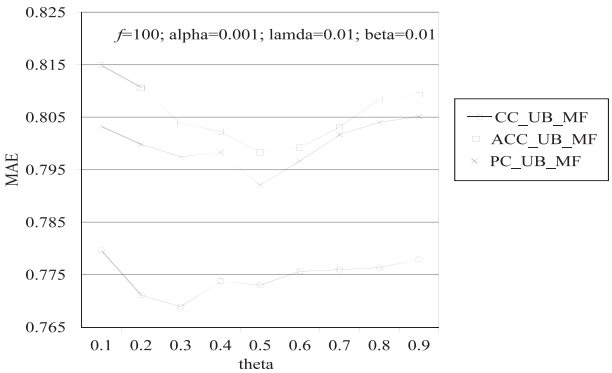

图 3　维度为 $l=12$,MAE 随 θ 变化曲线

(4)参数 β 对算法的影响。

在 UB_MF 算法中,参数 β 扮演着一个重要的角色,它决定着用户近邻模型对矩阵因子分解算法所起到的调节作用。接下来通过改变 β 参数来分析对文中提出算法推荐准确率的影响。设置学习速率 $\alpha=0.001$、$\lambda=\lambda_1=\lambda_2=0.01$、维度 $l=12$、$\theta=0.5$。在 3 种相似度计算方法下,得到的基于用户近邻约束的矩阵因子分解算法 MAE 随着阈值 $\beta(\beta=10^{-6},10^{-5},10^{-4},10^{-3},10^{-2},10^{-1},10^{0})$ 变化曲线如图 4 所示。

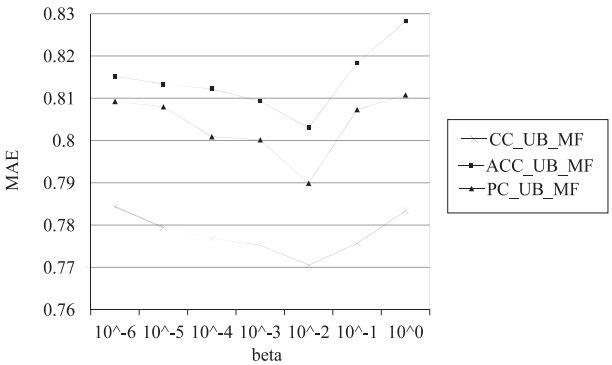

图 4　MAE 随 β 的变化曲线

从实验结果可以得出,随着 β 值的增加,MAE 的值首先下降,但是当 β 的值超过一定的阈值时,MAE 的值随着 β 的增加而上升。选择合适的 β 值可以有效地将用户评分矩阵和用户近邻进行结合产生更好的推荐,文中在 ML100k 数据集上进行实验,得到 β 最优取值为 10^{-2}。

4　结束语

随着电子商务网站的用户数量和商品数量的不断增加,用户-项目评分矩阵越来越大,同时数据也越来越稀疏,如何解决数据稀疏性和推荐效率问题显得尤其重要。矩阵因子分解算法不仅能很好地解决大的数据集问题,也能抽取出潜在的特征因子产生很好的推荐效果,因此得到了学者越来越多的关注和研究。文中在传统矩阵因子分解算法基础上,提出基于用户近邻约束的矩阵因子推荐算法。该算法利用矩阵分解降维技术和用户近邻模型,提高了算法推荐的实时性和预测的准确性。实验结果表明,基于近邻约束的矩阵因子分解算法在数据极端稀疏的情况下比传统矩阵因子分解算法可以获得更好的推荐质量。

文中提出的算法在不同的测试标准及不同数据集上的执行效果还需作进一步的研究。此外,矩阵分解模型中 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 矩阵第一次初始化赋值以及如何防止过度拟合问题,都值得进一步研究。

参考文献:

[1] Ricci F, Rokach L, Shapira B. Introduction to recommender systems handbook [M]//Recommender systems handbook. US: Springer, 2011.

[2] Ekstrand M D, Riedl J T, Konstan J A. Collaborative filtering recommender systems[J]. Foundations and Trends in Human-computer Interaction, 2011, 4(2): 81-173.

[3] Breese J S, Heckerman D, Kadie C. Empirical analysis of predictive algorithms for collaborative filtering[C]//Proceedings of the fourteenth conference on uncertainty in artificial intelligence. [s. l.]: Morgan Kaufmann Publishers Inc, 1998: 43-52.

[4] Jin R, Chai J Y, Si L. An automatic weighting scheme for collaborative filtering[C]//Proceedings of the 27th annual international ACM SIGIR conference on research and development in information retrieval. Sheffield: ACM, 2004: 337-344.

[5] Deshpande M, Karypis G. Item-based top-n recommendation algorithms[J]. ACM Transactions on Information Systems, 2004, 22(1): 143-177.

[6] Linden G, Smith B, York J. Amazon. com recommendations: item-to-item collaborative filtering[J]. IEEE Internet Computing, 2003, 7(1): 76-80.

[7] Ali K, van Stam W. TiVo: making show recommendations using a distributed collaborative filtering architecture[C]//Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining. Seattle: ACM, 2004: 394-401.

[8] Goldberg K, Roeder T, Gupta D, et al. Eigentaste: a constant time collaborative filtering algorithm[J]. Information Retrieval, 2001, 4(2): 133-151.

[9] Paterek A. Improving regularized singular value decomposition for collaborative filtering[C]//Proceedings of KDD cup and workshop. [s. l.]: [s. n.], 2007: 5-8.

[10] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems[J]. Computer, 2009, 42(8): 30-37.

[11] Mnih A, Salakhutdinov R. Probabilistic matrix factorization [C]//Advances in neural information processing systems. [s. l.]: [s. n.], 2007: 1257-1264.

(下转第 15 页)

粒子群优化算法、遗传算法的实验数据进行比较。可明显看出,文中改进算法的各项评价指标都比基本的粒子群算法、遗传算法的要更优。因此,基于改进的 PSOABC 与 K 均值聚类算法生成测试用例的效果明显比较好。

5　结束语

实验结果表明,以平均迭代次数和平均运行时间作为评价指标,文中基于改进的 PSOABC 与 K 均值聚类算法与基本的粒子群优化算法、遗传算法的实验数据进行比较,可明显看出改进算法的各项评价指标都比基本的粒子群算法、遗传算法的要更优。由此可见,基于改进的 PSOABC 与 K 均值聚类算法的测试用例自动生成方法具有测试用例自动生成效率高、收敛能力强等优点,且有一定的实用性。

参考文献:

[1] Eberhart R C, Kennedy J. A new optimizer using particle swarm theory[C]//Proc of the sixth international symposium on micro machine and human science. Nzgoya, Japan: [s. n.],1995:39-43.

[2] Kennedy J, Eberhart R C. Particle swarm optimization[C]//Proc of IEEE conf on neural network. Perth: IEEE, 1995: 1942-1948.

[3] Kennedy J, Eberhart R C. Swarm intelligence[M]. San Francisco, CA: Morgan Kaufmann Publishers Inc, 2001.

[4] 郭长友. 一种自适应惯性权重的粒子群优化算法[J]. 计算机应用与软件,2011,28(6):289-292.

[5] Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony algorithm [J]. Journal of Global Optimization, 2007, 39(3): 459-471.

[6] Karaboga D. An idea based on honey bee swarm for numerical optimization[D]. Ercyes: Ercyes University, 2005.

[7] Karaboga D, Basturk B. A comparative study of artificial bee colony algorithm[J]. Applied Mathematics and Computation, 2009, 214(1): 108-132.

[8] 刘　路, 王太勇. 基于人工蜂群算法的支持向量机优化[J]. 天津大学学报, 2011, 44(9): 803-809.

[9] 毕晓君, 王艳娇. 改进人工蜂群算法[J]. 哈尔滨工程大学学报, 2012, 33(1): 117-123.

[10] 谢秀华, 李陶深. 一种基于改进 PSO 的 K-means 优化聚类算法[J]. 计算机技术与发展, 2014, 24(2): 34-38.

[11] 杨　韬, 邵良杉. 采用改进的 k 均值聚类分析策略的粒子群算法[J]. 计算机工程与应用, 2009, 45(12): 52-54.

[12] 欧陈委. K-均值聚类算法的研究与改进[D]. 长沙: 长沙理工大学, 2011.

[13] 周爱武, 陈宝楼, 王　琰. K-Means 算法的研究与改进[J]. 计算机技术与发展, 2012, 22(10): 101-104.

[14] 傅景广, 许　刚, 王裕国. 基于遗传算法的聚类分析[J]. 计算机工程, 2004, 30(4): 122-124.

(上接第 5 页)

[12] 罗　辛, 欧阳元新, 熊　璋, 等. 通过相似度支持度优化基于 K 近邻的协同过滤算法[J]. 计算机学报, 2010, 33(8): 1437-1445.

[13] Anand D, Bharadwaj K K. Utilizing various sparsity measures for enhancing accuracy of collaborative recommender systems based on local and global similarities[J]. Expert Systems with Applications, 2011, 38(5): 5101-5109.

[14] Zhou K, Yang S H, Zha H. Functional matrix factorizations for cold-start recommendation[C]//Proceedings of the 34th international ACM SIGIR conference on research and development in information retrieval. [s. l.]: ACM, 2011: 315-324.

[15] Cergani E, Miettinen P. Discovering relations using matrix factorization methods[C]//Proceedings of the 22nd ACM international conference on information & knowledge management. [s. l.]: ACM, 2013: 1549-1552.

[16] Zhang Y, Zhu X, Shen Q. A recommendation model based on collaborative filtering and factorization machines for social networks[C]//Proc of 5th IEEE international conference on broadband network & multimedia technology. [s. l.]: IEEE, 2013: 110-114.

[17] Yu K, Zhu S, Lafferty J, et al. Fast nonparametric matrix factorization for large-scale collaborative filtering[C]//Proceedings of the 32nd international ACM SIGIR conference on research and development in information retrieval. [s. l.]: ACM, 2009: 211-218.

[18] Takács G, Pilászy I, Németh B, et al. Scalable collaborative filtering approaches for large recommender systems[J]. Journal of Machine Learning Research, 2009, 10(3): 623-656.

[19] 李　改, 李　磊. 基于矩阵分解的协同过滤算法[J]. 计算机工程与应用, 2011, 47(30): 4-7.

[20] Takács G, Pilászy I, Németh B, et al. Investigation of various matrix factorization methods for large recommender systems [C]//Proc of IEEE international conference on data mining workshops. [s. l.]: IEEE, 2008: 553-562.

[21] 杨　阳, 向　阳, 熊　磊. 基于矩阵分解与用户近邻模型的协同过滤推荐算法[J]. 计算机应用, 2012, 32(2): 395-398.

基于用户近邻约束的矩阵因子分解算法

作者： 查九， 李振博， 徐桂琼， ZHA Jiu， LI Zhen-bo， XU Gui-qiong
作者单位： 上海大学 管理学院,上海,200444
刊名： 计算机技术与发展 ISTIC
英文刊名： Computer Technology and Development
年，卷(期)： 2015(6)