第46卷第2期 中国科学技术大学学报 Vol. 46, No. 2
2016年2月 JOURNAL OF UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA Feb. 2016

文章编号:0253-2778(2016)02-0113-07

基于用户及物品间差异的推荐算法研究

李 强,何兴盛,傅忠谦

(中国科学技术大学电子科学与技术系,安徽合肥 230027)

摘要:推荐系统是解决信息过载问题最有效的工具之一,协同过滤是目前推荐算法中广泛应用的技术,然而协同过滤算法存在着诸如数据稀疏、难以扩展等问题.在基于偏好算法的基础上,通过把用户评分按照用户评分偏好和物品得分趋势分类,在每类上进行线性回归,得到了基于用户及物品间差异的回归模型.该模型不仅能改善数据稀疏和可扩展性问题,而且能够降低计算复杂度和空间复杂度.实验结果表明改进后的算法在近似的计算复杂度情况下,预测精度比基于偏好算法平均提高了3.97%.

关键词:推荐系统;协同过滤;用户评分偏好;物品得分趋势

中图分类号:TP391 **文献标识码**:A doi:10.3969/j.issn.0253-2778.2016.02.00

引用格式:Li Qiang, He Xingsheng, Fu Zhongqian. Recommender algorithms based on tendencies between users and items[J]. Journal of University of Science and Technology of China, 2016,46(2):113-119.
李强,何兴盛,傅忠谦. 基于用户及物品间差异的推荐算法研究[J]. 中国科学技术大学学报,2016,46(2):113-119.

Recommender algorithms based on tendencies between users and items

LI Qiang, HE Xingsheng, FU Zhongqian

(*Department of Electronic Science and Technology, University of Science and Technology of China, Hefei* 230027, *China*)

Abstract: Recommender systems are one of the most effective technologies to help users filter the overload of information, and collaborative filtering (CF) is one of the most widely used techniques in recommender systems. However, CF algorithms have difficulty dealing with the problems such as the sparseness of data and the scalability for new users. As an alternative, an improved algorithm based on the preference (tendencies-based, TB) algorithm was proposed. In the proposed method: firstly, the user rating set was classified into different groups according to user rating preference and item rated tendencies; then the regression model was obtained by linear regression performed on each class. The improved model not only achieves higher accuracy in rating prediction on sparse datasets, but also greatly reduces computational complexity and space complexity. Through extensive experiments on three benchmark data sets, the results show that the improved approach increases recommendation accuracy by an average of 3.97% compared with TB algorithm.

Key words: recommender system; collaborative filtering; user rating tendency; item rated tendency

收稿日期:2015-11-23;**修回日期**:2016-02-25

作者简介:李强,男,1990年生,硕士生. 研究方向:推荐系统. E-mail: liqianai@mail.ustc.edu.cn

通讯作者:傅忠谦,副教授. E-mail: zqfu@ustc.edu

0 引言

随着互联网近年来爆炸式的发展,大量信息被呈现在用户面前.用户面对如此多的信息,如何从中找到自己感兴趣的信息是一件非常困难的事情.信息爆炸使得信息的利用率反而降低,这种现象被称为信息超载[1].个性化推荐是帮助用户寻找信息、克服信息超载的重要工具.推荐系统主要是从用户的交易记录和评分记录等历史行为信息出发[2],建立数学模型来挖掘用户的兴趣和需求,从而为用户做出个性化推荐.在过去20年里,学术界对推荐系统做了大量的探索研究,并提出了很多算法:协同过滤[3]、谱分析法[4]、隐因子模型[5]、混合推荐[6].推荐系统已经被广泛应用在电子商务以及新兴的web应用中,如淘宝、Amazon、社交网络(Facebook和微博)等都采用了推荐系统为用户提供个性化推荐服务.

由于算法的计算复杂度低和算法思想的简单直观,协同过滤技术得到了广泛的关注和研究.协同过滤思想由最初由Glodberg等于1992年首次提出,其核心思想是:对于某位用户,如果其他邻居用户与他有相似的兴趣喜好,那么邻居用户喜欢的物品,该用户也会喜欢.然而,协同过滤算法依然面临着诸如数据稀疏、难以扩展等问题.尽管很多改进的算法被提出以期解决稀疏性、扩展性问题:例如用来解决扩展性问题的基于奇异值分解的矩阵分解法[7-9]和基于二分网络的物理扩散过程算法[10-12],以及解决数据稀疏问题的主成分分析法[13];但这些问题并没有从根本上解决.

Cacheda等[14]在2011年提出了一种新的协同过滤方法——基于用户评分偏好和物品得分趋势(tendencies based, TB)的推荐算法.TB算法涉及4个特征变量,分别为用户的平均分、物品的平均得分、用户的评分偏好以及物品的得分趋势.该算法使用较少的计算量便可有效解决推荐系统中协同过滤算法存在的扩展性问题,适合应用在大型的推荐系统中.本文受基于用户偏好和商品质量算法(algorithm based on item quality and user rating preferences,简称QP算法)[2]启发,在TB算法基础上,通过分析算法所用的数据集,提出了对用户的平均分、物品的平均得分、用户的评分偏好以及物品的得分趋势4个变量线性回归可得到用户对物品的评分预测模型的假设,对TB算法做了改进,将改进后的算法称之为基于偏好的回归(tendencies-based regression,TR)算法.在3个数据集上进行了实验仿真,结果表明改进后的算法相比于基于用户和物品的协同过滤不仅能改善数据稀疏和可扩展性问题,并且能够降低计算复杂度和空间复杂度,较低的计算复杂度以及可扩展性适用于大型推荐系统;同时,TR算法比TB算法有更高的准确率.

1 基于相似度的协同过滤算法以及TB算法

很多推荐问题可以归结为预测评分问题.给定1个无时间信息的用户显性反馈数据集或用户评分数据集 $D=\{(\alpha,i,r)\}$,其中每3个元组(α,i,r)代表了1个用户α对1个物品i评了r分.如果有N位用户和M件物品,评分数据集可以建立1个2维稀疏矩阵 $R\in R^{N\times M}$.其中每1行代表1个用户对所有物品的评分,每1列代表1个物品被所有用户的评分.由于评分矩阵R是稀疏的,矩阵中含有大量的缺失值,因此推荐系统评分预测问题的任务就是如何通过观测值来补全缺失值.文中出现的符号的含义在表1中给出,分别用希腊字母和拉丁字母表示用户和物品的索引号.

表1 符号所代表的含义

Tab.1 The notations of symbols

符 号	符号含义
$r_{\alpha,i}$	用户α对物品i的评分
$p_{\alpha,j}$	预测用户α对物品j的评分
U	系统中所有用户的集合
I	系统中所有物品的集合
U_i	对物品i评过分的用户集合
$\lvert U_i\rvert$	对物品i评过分的用户集合数目
I_α	用户α评过分的物品集合
$\lvert I_\alpha\rvert$	用户α评过分的物品集合数目
U_α	与用户α相似的用户集合
I_i	与物品i相似的物品集合
$\overline{r_\alpha}$	用户α的评分的平均分
$\overline{r_i}$	物品i的平均得分
τ_α	用户α的评分趋势(偏好)
τ_i	物品i的得分趋势

1.1 基于相似度的协同过滤算法

协同过滤(CF)是推荐系统领域最著名的算法.按照用户和物品的相似度,CF算法可分为基于用户的协同过滤算法(user-based collaborative filtering, UCF)[15]和基于物品的协同过滤算法

(item-based collaborative filtering, ICF)[16]. CF 算法的核心思想可分为两步骤:首先,是利用用户的历史评分信息计算用户之间的相似性;然后,利用与目标用户相似性较高的邻居对特定产品的评分来预测目标用户对该特定产品的评分.

UCF 预测用户对某件物品的评分是基于同用户有相似兴趣的邻居用户的评分,因此计算用户之间的相似度是该类算法的关键. Pearson 相关性是计算用户之间相似度常用的方法,用户 α 和 β 之间的 Pearson 相似度定义为

$$s(\alpha,\beta)=\frac{\sum_{i\in I_\alpha\cap I_\beta}(r_{\alpha,i}-\overline{r_\alpha})(r_{\beta,i}-\overline{r_\beta})}{\sqrt{\sum_{i\in I_\alpha\cap I_\beta}(r_{\alpha,i}-\overline{r_\alpha})^2\sum_{i\in I_\alpha\cap I_\beta}(r_{\beta,i}-\overline{r_\beta})^2}} \tag{1}$$

由式(1)可知,$s(\alpha,\beta)$取值范围为$[-1,1]$,当两个用户的兴趣完全一致时得到 $s(\alpha,\beta)=1$.

在得到评分活跃的用户之间的相似度后,预测用户 α 对物品 i 的评分的表达式定义为

$$p_{\alpha,i}=\overline{r_\alpha}+\frac{\sum_{\beta\in U_\alpha}s(\alpha,\beta)(r_{\beta,i}-\overline{r_\beta})}{\sum_{\beta\in U_\alpha}|s(\alpha,\beta)|} \tag{2}$$

与 UCF 类似,ICF 算法首先计算物品间的相似度,物品 i 和 j 之间的相似度可通过以下公式计算得到:

$$s(i,j)=\frac{\sum_{\alpha\in U_i\cap U_j}(r_{\alpha,i}-\overline{r_\alpha})(r_{\alpha,j}-\overline{r_\alpha})}{\sqrt{\sum_{\alpha\in U_i\cap U_j}(r_{\alpha,i}-\overline{r_\alpha})^2\sum_{\alpha\in U_i\cap U_j}(r_{\alpha,j}-\overline{r_\alpha})^2}} \tag{3}$$

由式(3)可知,$s(i,j)$的取值范围为$[-1,1]$.

同理在得到各商品之间的相似度之后,用户 α 对物品 i 的评分可以用下式预测:

$$p_{\alpha,i}=\frac{\sum_{j\in I_i}(s(i,j)\ r_{\alpha,j})}{\sum_{j\in I_i}|s(i,j)|} \tag{4}$$

1.2　TB 算法

TB 算法的核心是用户或者物品之间的差异性. 这种差异性在 TB 算法中被称之为用户的评分和物品的得分趋势(下文简称趋势). 与寻找用户相似度的算法相比,得到趋势只需很少的计算时间和内存以及数据.

用户的评分偏好可以被理解为用户的倾向,倾向给物品正面评价或负面评价. 某用户倾向于给物品的评分低于该物品的平均分,这位用户便是倾向给负面评价($\tau_\alpha<0$). 所以,用户的评分偏好(τ_α)定义为下式:

$$\tau_\alpha=\frac{\sum_{i\in I_\alpha}(r_{\alpha,i}-\overline{r_i})}{|I_\alpha|} \tag{5}$$

一位用户评价了该物品,如果该物品的评分高于这位用户评价过的所有物品的评分,则其得分趋势是正面的($\tau_i>0$). 物品的得分趋势可由下式计算得到:

$$\tau_i=\frac{\sum_{\alpha\in U_i}(r_{\alpha,i}-\overline{r_\alpha})}{|U_i|} \tag{6}$$

TB 算法使用了$\overline{r_\alpha}$,$\overline{r_i}$,τ_α,τ_i 4 个变量做评分预测,预测模型按这 4 个变量可分为 4 种情形:①$\tau_\alpha>0$ 且 $\tau_i>0$;②$\tau_\alpha<0$ 且 $\tau_i<0$;③$(\tau_\alpha-\tau_i)\cdot(\overline{r_\alpha}-\overline{r_i})>0$;④$(\tau_\alpha-\tau_i)\cdot(\overline{r_\alpha}-\overline{r_i})<0$. 4 种情况的预测模型可归纳为

$$\left.\begin{aligned}&p_{\alpha,i}=\max(\overline{r_\alpha}+\tau_i,\overline{r_i}+\tau_\alpha),\\&p_{\alpha,i}=\min(\overline{r_\alpha}+\tau_i,\overline{r_i}+\tau_\alpha),\\&p_{\alpha,i}=\\&\min(\max(\overline{r_\alpha},\mu(\overline{r_i}+\tau_\alpha)+(1-\mu)(\overline{r_\alpha}+\tau_i)),\overline{r_i}),\\&p_{\alpha,i}=\mu\overline{r_i}+(1-\mu)\overline{r_\alpha}.\end{aligned}\right\} \tag{7}$$

2　改进的算法

经典的协同过滤方法都是基于计算用户或者物品间的相似性,其中很多算法用来处理用户(物品)数据信息,寻找隐藏的用户、物品关系. 但是这些算法需要大量的数据信息和计算量来计算用户之间的相似矩阵,在用户-物品评分矩阵稀疏时,算法的预测精度会有所下降. 兴趣相似的用户评分方式也有可能存在差异:有的用户倾向于给高分,给其认为差的物品低分;而有的用户倾向于给低分,给其认为最好的物品高分. 用户之间的评分差异与用户的品味和兴趣是相关的,但这不是决定评分的唯一因素. 物品的质量也会影响其得分:质量好的物品会有得分较高的趋势,而质量低的物品会有得分较低的趋势,物品的得分趋势可以用来体现物品质量的差异. 所以用户和物品会同时影响用户给物品评分. 从式(5)知用户的评分偏好是该用户对其评价过物品的评分与这些物品平均分的平均偏差,同理在式(6)中物品

的得分趋势为该物品的得分与评价过该物品的用户平均分的平均偏差，刻画了用户及物品间的差异，可以看作个性化因素. 由此，在预测用户 α 对物品 i 的打分时，本文提出将用户 α 和物品 i 平均分作为基础打分，用户的评分偏好和物品得分趋势作为用户 α 和物品 i 的个性化的方法：

$$p_{\alpha,i} = A\overline{r_\alpha} + B\tau_\alpha + C\overline{r_i} + D\tau_i \tag{8}$$

对 TB 算法进行了改进，式中 $\overline{r_\alpha}$，$\overline{r_i}$，τ_α，τ_i 分别为用户 α 和物品 i 的平均分、用户 α 的评分偏好、物品 i 的得分趋势. A，B，C，D 为 4 个变量的系数，可通过线性回归得到.

改进后的算法仍然是基于用户和物品的趋势，但与 TB 算法不同的是，该方法通过对 4 个变量训练回归得到预测模型. TR 算法同 TB 算法一样将数据集 τ_α，τ_i，$\overline{r_\alpha}$，$\overline{r_i}$ 分 4 种情形：① $\tau_\alpha>0$ 且 $\tau_i>0$；② $\tau_\alpha<0$ 且 $\tau_i<0$；③ $(\tau_\alpha-\tau_i)\cdot(\overline{r_\alpha}-\overline{r_i})>0$；④ $(\tau_\alpha-\tau_i)\cdot(\overline{r_\alpha}-\overline{r_i})<0$. 然后分别对这 4 种情况训练回归模型，得到每组情况中 4 个变量的回归系数 A，B，C，D（分别是含有 4 个元素的列向量）. 此方法中的用户的平均分、物品的平均得分、用户的评分偏好以及物品的得分趋势通过简单和少量的计算便可到. 在预测得到评分矩阵中的缺失值后，便可把得分较高的物品推荐给用户. TR 的预测模型可概述为

$$\left.\begin{aligned} p_{\alpha,i} &= A[1]\overline{r_\alpha} + B[1]\tau_i + C[1]\overline{r_i} + D[1]\tau_\alpha,\\ p_{\alpha,i} &= A[2]\overline{r_\alpha} + B[2]\tau_i + C[2]\overline{r_i} + D[2]\tau_\alpha,\\ p_{\alpha,i} &= A[3]\overline{r_\alpha} + B[3]\tau_i + C[3]\overline{r_i} + D[3]\tau_\alpha,\\ p_{\alpha,i} &= A[4]\overline{r_\alpha} + B[4]\tau_i + C[4]\overline{r_i} + D[4]\tau_\alpha \end{aligned}\right\} \tag{9}$$

TR 算法具体描述如下：

输入：训练集数据 $T=\{(\text{user\_id},\text{item\_id},\text{rate})\}$，其中 user_id 为用户编号，item_id 为物品编号，rate 为用户对物品的评分(1～5)；

输出：测试集中用户 β 对物品 j 的预测评分 $\{p_{\beta,j}\mid j\in I-I_\beta\}$.

(a)在训练集中，计算用户的评分平均值 $\overline{r_\alpha}$，再根据式(5)计算用户的评分偏好 τ_α；计算每个物品的得分均值 $\overline{r_i}$，再根据式(6)计算物品的得分趋势 τ_i；

(b)将训练集按 τ_α，τ_i，$\overline{r_\alpha}$，$\overline{r_i}$ 分 4 种情形，分别做回归训练，通过式 $[A,B,C,D]=\text{argmin}(r_{\alpha,i}-p_{\alpha,i})^2$ 训练出变量系数 A，B，C，D；$p_{\alpha,i}$ 由式(8)计算得来.

(c)根据式(8)预测测试集中用户对物品的评分.

对于一个有 N 个用户、M 个物品规模的数据集，TR，TB 算法的计算复杂度是 $O(MN)$，UCF 和 ICF 算法的计算复杂度分别是 $O(MN^2)$，$O(NM^2)$. ICF 和 UCF 都是寻找用户或者物品之间的隐藏关系，需要计算和保存所有用户-用户、物品-物品之间的相似矩阵，需要 $O(N^2)$ 和 $O(M^2)$ 的空间复杂度和 $O(MN^2)$ 和 $O(NM^2)$ 的计算复杂度，而 TR 算法只需要 $O(2(M+N))$ 的空间复杂度. 在 3 组不同规模的数据集上，将训练集和测试集的比例设为9∶1，表 2 对比列出了 4 种算法的时间.

表 2　3 组数据集上 4 种算法消耗时间(单位：s)

Tab. 2　The time of four algorithms on three datasets (unit: s)

算法/数据集	ML. A	Netflix	ML. B
UCF	35.8	536.0	3000.5
ICF	76.9	289.2	956.14
TR	1.2	3.4	11.5
TB	0.9	2.9	9.6

从表中可以看出 TR 和 TB 算法都有着很少的消耗时间，TR 算法的时间复杂度主要表现在线性回归计算上. 表 2 的数据计算平台为 Windows OS 的台式计算机（CPU：Intel Core 2.34 GHZ；Memory：4 GB）

3　实验结果

本文使用了 3 组常用的数据集，分别为 MovieLens[17] A (ML. A)，MovieLens B(ML. B)以及 Netflix[18]. 3 个数据集的评分都是从 1 分到 5 分，分别用 M 和 N 代表数据集中用户和电影数量，$|E|$ 代表已有的评分条数，数据集的稀疏度表示为 $|E|/(M*N)$. ML. A 是 MovieLens 中最小的数据集，包含了 943 个用户、1 682 部电影、100 000条评分记录，稀疏度为 6.30×10^{-2}；ML. B 是 MovieLens 中 6 040 名用户对 3 952 部电影的1 000 209条评分记录，稀疏度为 4.19×10^{-2}. MovieLens 数据集每名用户至少评价过 20 部电影. Netflix 是从 Netflix 比赛原始数据集随机抽取 3 000 名用户、3 000 部电影、292 881条评分记录组成的数据，每名用户同样至少评价过 20 部电影，稀疏度为 3.25×10^{-2}.

为了测试改进的算法的性能，我们把数据集 $|E|$ 随机分成训练集 E^T 和测试集 E^P（$E^T\cup E^P=E$，$E^T\cap E^P=\varnothing$）. E^T 用来训练测试模型，E^P 作为预测未知评分的原始数据. 在实验中，训练集由从数据集 E 中随机抽取的 $p\%$ 组成，剩下的 $(100-p)\%$ 作为测试集，可以通过调整参数 p 控制实验中数据的稀疏

度.本文将 p 设成从 10 以间隔 10 递增到 90 来测试算法在不同稀疏度下的性能.

有很多指标被用来评估推荐系统的性能[3],平均绝对误差 MAE(mean absolute error)和平均差方根 RMSE(root mean square error)[3]是两个最常用的衡量预测精度的标准.关于 RMSE 和 MAE 这两个指标的优缺点,Netflix 认为 RMSE 加大了对预测不准的用户物品评分的惩罚(平方项的惩罚),因而对系统的评测更加苛刻.本文使用 RMSE 作为算法的衡量指标,其定义如下式:

$$\mathrm{RMSE}=\sqrt{\frac{\sum_{(a,i)\in E^P}(r_{a,i}-p_{a,i})^2}{|E^P|}} \tag{10}$$

在现实环境中,用户往往更倾向于评价自己喜欢的物品并给出较高的评分(如淘宝评价中好评数往往大于中评差评数).图 1 给出了 3 组数据集评分的分布直方图,这 3 组数据有着类似的分布,评分为 3,4,5 的数量比评分为 1,2 的更多,这符合实际情况,因为用户往往只会评价自己喜欢的电影[14].需要引起注意的是,改进的算法是依赖于这 3 个数据集的统计特性的,所以该算法的性能在别的数据集或者领域上还有待验证.除此之外,如评分矩阵的稀疏度、用户数和物品数的比例、评分的变化、数据规模等因素都可能影响算法的预测精度.

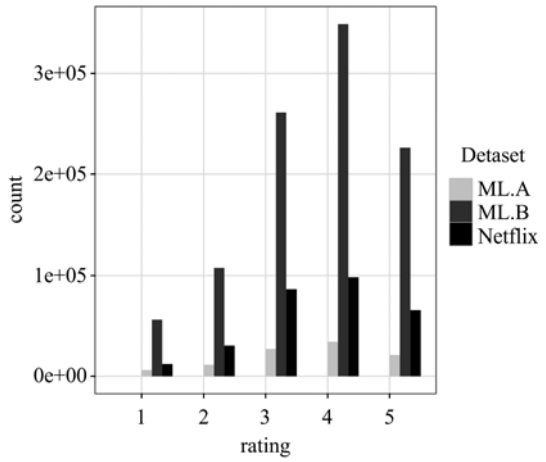

图 1　3 组数据的评分分布直方图

Fig. 1　The rate histogram of three datasets

由式(7)知,TB 算法的可调参数 μ 调节着物品平均分和用户平均分对用户评分的贡献权重关系.当 $\mu=0$ 时,TB 算法退化为由用户平均分来主导预测用户对物品的评分;当 $\mu=1$ 时,TB 算法退化为由物品平均分来主导预测用户对物品的评分.图 2 给出了可调参数 μ(取值从 0 以 0.1 递增到 1)不同取值时的 RMSE 对比情况.从图中可知,当 μ 分别为 0.9,0.7,0.7 时,数据集 ML.A,Netflix,ML.B 的 RMSE 在 μ 的取值范围内为最优.

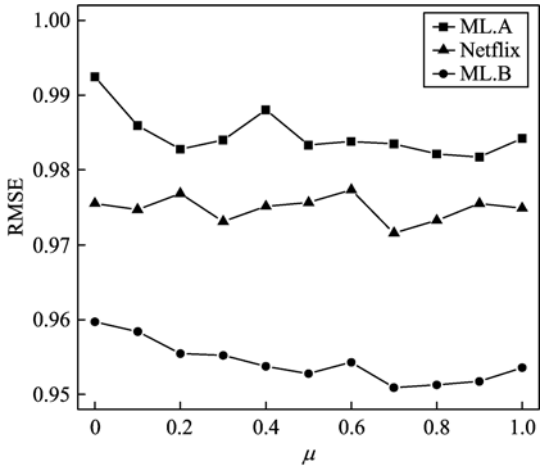

图 2　3 组数据集 μ 不同取值的 RMSE 对比

Fig. 2　RMSE comparison with different μ on three datasets

本文将 TR 算法与传统的 UCF 和 ICF 以及 TB 算法进行了对比.其中 UCF 和 ICF 算法需要从其他用户(物品)选取 K 个与当前用户(物品)相似度最高的用户(物品)作为最近邻,实验过程中,我们将近邻数 K 从 10 递增到 200,间隔为 10,当 $K=30$ 时,UCF 和 ICF 算法到达最优,所以我们选择 $K=30$ 作为最近邻居数.TB 算法中的可调参数 μ 设为图 2 中最优值的点:ML.A 数据集设为 0.9,ML.B 和 Netflix 都设为 0.7.图 3 展示了在不同训练集的比例下各个算法的表现情况,图中横坐标代表了训练集的比例,纵坐标是评估算法性能的指标 RMSE.从图 3 中可得知,在 3 组数据上 TR 算法的性能比其他 3 种算法表现都好,并且 RMSE 在训练数据集达到 60%之后就趋于平稳,这意味着 TR 算法需要较少的训练数据便可得到模型,这一特性适于应用到实际推荐系统中.表 3 给出了 TR 算法相比于 TB 性能提升的具体结果,在 3 组数据集上分别平均提升了 3.12%,4.30%,4.50%.

4　结论

基于用户的评分偏好和物品的得分趋势的 TR 算法是一种简单但有效的方法,预测用户的评分结果也是较好的.实验结果表明:根据用户偏好和物品偏好数据分类,然后分别对用户平均分、物品平均分、用户偏好、物品得分趋势线性回归可以得到用户

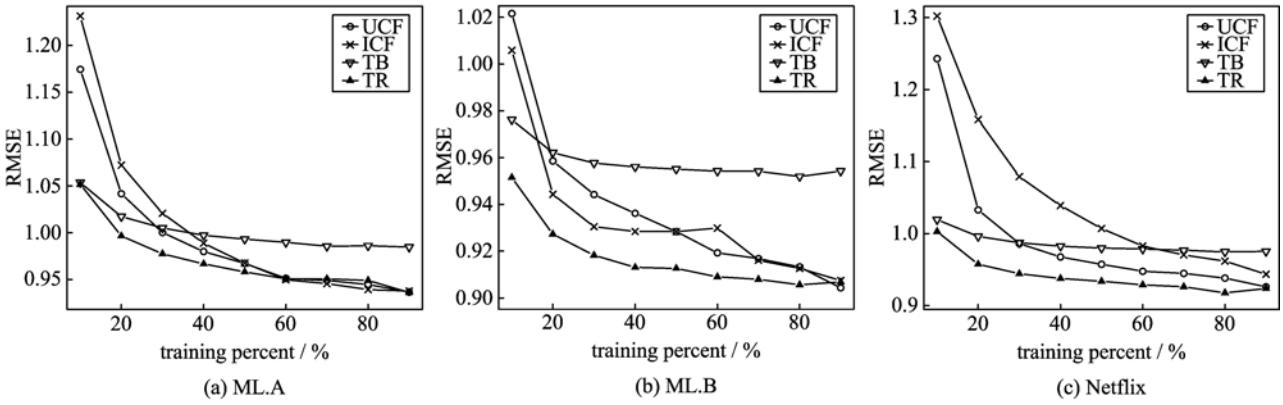

图 3 TR 与 TB,UCF,ICF 的性能指标(RMSE)对比

Fig. 3 The accuracy of TR compared with TB, UCF, ICF algorithms

表 3 TR 相比 TB 算法的 RMSE 指标提升

Tab. 3 The RMSE improvement of TR compared with TB on ML. A, ML. B, and Netflix datasets

Training percent/%	RMSE								
	ML. A			ML. B			Netflix		
	TB	TR	Improvement/%	TB	TR	Improvement/%	TB	TR	Improvement/%
10	1.054	1.051	0.28	0.976	0.951	2.56	1.019	1.002	1.67
20	1.017	0.996	2.06	0.962	0.927	3.63	0.995	0.957	3.82
30	1.005	0.977	2.79	0.957	0.918	4.07	0.987	0.944	4.36
40	0.997	0.966	3.17	0.956	0.913	4.49	0.982	0.937	4.58
50	0.993	0.958	3.52	0.955	0.912	4.50	0.979	0.933	4.69
60	0.989	0.950	3.94	0.954	0.909	4.71	0.978	0.928	5.11
70	0.986	0.950	3.65	0.954	0.908	4.82	0.976	0.926	5.12
80	0.986	0.949	3.75	0.952	0.905	4.93	0.974	0.917	5.85
90	0.984	0.936	4.88	0.954	0.906	5.03	0.975	0.923	5.33
Ave_Improvement/%		3.12			4.30			4.50	

评分预测模型,该模型(TR)比 TB 的预测精度(RMSE)平均提升了 3.97%. TR 算法在训练数据集为 60%的情况下预测性能就趋于稳定,能够改善数据稀疏和可扩展性问题,并且降低了时间复杂度和空间复杂度,这些优点适合于应用在实际的推荐系统中.但是 TB 算法性能的提升是基于所用数据集(电影领域)的特性的,在其他领域的效果还有待考证.本文在预测用户评分时并未考虑时间因素对用户的影响,训练集和测试集中的评分没有时间前后之分,所以在预测时并不能肯定是否用了历史评分信息去预测未来的评分,在以后的工作中将会考虑这种时间先后因素;此外,用户的兴趣是随着时间变化的,今后将研究这种变化对预测评分的影响.

参考文献(References)

[1] 刘建国, 周涛, 汪秉宏. 个性化推荐系统的研究进展[J]. 自然科学进展,2009,19(1):1-15.

[2] Guan Y, Cai S M, Shang M S. Recommendation algorithm based on item quality and user rating preferences[J]. Frontiers of Computer Science, 2014, 8: 289-297, 2014.

[3] Herlocker J L, Konstan J A, Terveen L G, et al. Evaluating collaborative filtering recommender systems [J]. ACM Transactions on Information Systems, 2004, 22: 5-53.

[4] Ren J, Zhou T, Zhang Y C. Information filtering via self-consistent refinement [J]. EPL (Europhysics Letters), 2008, 82: 58007; doi: 10.1209/0295-5075/82/58007.

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

[6] Good N, Schafer J B, Konstan J A, et al. Combining collaborative filtering with personal agents for better recommendations[C]// Proceedings of the sixteenth national conference on Artificial intelligence and the

eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence. Menlo Park, CA, USA : American Association for Artificial Intelligence, 1999: 439-446.

[7] Dror G, Koenigstein N, Koren Y. Web-scale media recommendation systems[J]. Proceedings of the IEEE, 2012,100: 2 722-2 736.

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

[9] Yang Yang, Xiang Yang, Xiong Lei. Collaborative filtering and recommendation algorithm based on matrix factorization and user nearest neighbor model [J]. Journal of Computer Applications, 2012, 32: 395-398.
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: 395-398.

[10] Zhou T, Ren J, Medo M, et al. Bipartite network projection and personal recommendation[J]. Physical Review E,2007, 76: 046115;doi: 10.1103/PhysRevE.76.046115.

[11] He X S, Zhou M Y, Zhuo Z, et al. Predicting online ratings based on the opinion spreading process[J]. Physica A: Statistical Mechanics and its Applications, 2015, 436:658-664.

[12] Zhou T, Kuscsik Z, Liu J G, et al. Solving the apparent diversity-accuracy dilemma of recommender systems[J]. Proceedings of the National Academy of Sciences, 2010, 107: 4 511-4 515.

[13] Goldberg K, Roeder T, Gupta D, et al. Eigentaste: A constant time collaborative filtering algorithm [J]. Information Retrieval, 2001, 4: 133-151.

[14] Cacheda F, Carneiro V, Fernández D, et al. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems[J]. ACM Transactions on the Web, 2011, 5(1):161-171

[15] Resnick P, Iacovou N, Suchak M, et al. GroupLens: An open architecture for collaborative filtering of netnews [C]// Proceedings of the 1994 ACM conference on Computer supported cooperative work. New York, NY, USA: ACM, 1994: 175-186.

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

[17] Harper F M, Konstan J A. The MovieLens Datasets: History and Context [J]. ACM Transactions on Interactive Intelligent Systems (TiiS), 2015, 5(4): 19;doi: 10.1145/2827872.

[18] Bennett J, Lanning S. Thenetflix prize [C]// Proceedings of KDD cup and workshop. 2007: 35.