基于上下文项目评分分裂的协同过滤推荐

何 明 刘 毅 常盟盟 吴小飞

(北京工业大学计算机学院 北京 100124)

摘 要 上下文感知推荐系统的主要任务是利用上下文信息进一步提高推荐系统的推荐精度和用户满意度。提出了一种基于上下文项目评分分裂的推荐方法。该方法首先依据项目分裂判别标准对多维度上下文信息下的项目进行分裂,然后根据分裂结果并通过上下文维度进行聚类。在此基础上,利用协同过滤推荐算法进行未知评分预测。最后,面向不同的项目分裂标准,在LDOS-CoMoDa数据集上进行仿真对比实验。实验结果表明,相对于其他推荐算法,该方法有效提升了推荐精度,达到了提高推荐质量效果的目的。

关键词 上下文感知系统推荐,基于项目的上下文分裂方法,协同过滤,聚类

中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.03.051

Collaborative Filtering Recommendation Based on Item Splitting

HE Ming LIU Yi CHANG Meng-meng WU Xiao-fei

(College of Computer Science, Beijing University of Technology, Beijing 100124, China)

Abstract Context-aware recommendation system is an effective way to improve the recommendation accuracy and user satisfaction by using context information. In this paper, an efficient context-item splitting approach for context-aware recommendation was proposed. Firstly, the items are divided according to the item split criterion. Secondly, the clustering is carried out through the context dimension based on the splitting results. Thirdly, the collaborative filtering recommendation algorithm is used to predict the unknown ratings. Finally, simulation experiments are conducted on the LDOS-CoMoDa data set for different splitting criteria. The experimental results demonstrate that this method can effectively improve the accuracy of the recommendation and achieve the goal of improving the quality of recommendation.

Keywords Context-aware recommendation, Item-splitting context-aware approaches, Collaborative recommendation, Cluster

1 引言

随着互联网爆炸式发展,"信息过载"问题日益突出。搜索引擎虽然在很大程度上缓解了过载问题,但仍不能满足日益丰富的个性化需求[1]。传统推荐系统挖掘用户与项目(user-item)之间的二元关系,为用户推荐可能满足需求的项目。其中基于协同过滤算法构建的推荐系统最具代表性,作为一种独立的方法,通过计算用户提供项目评分的历史日志记录来形成推荐。实际上,用户的兴趣一般都相对稳定,但对于一个具体项目而言,一些额外的因素会严重影响它的评分。在某些消费领域,同样的项目会因为在不同的上下文信息中导致差异很大[2]。比如,对于一次去沙滩的旅行,在冬天和夏天的感受是完全不一样的,然而,传统的协同过滤推荐系统却不能区分这两种不同的体验,因此可能会提供较低的旅游推荐质量。

上下文感知推荐系统(Context-Aware Recommender Systems, CARS)[3-4]是一个崭新的研究领域,由Adomavicius和Tuzhilin等人较早提出。他们认为将上下文信息融入到推荐系统中有利于提高推荐系统的精度[3]。Dey等人将上下文定义为"上下文是能够用来描绘事物任何情况信息的实体"[5],实体指的是能够描述用户和项目状态的上下文变量,是一种经验。CARS将标准的"user-item"二元关系进行了拓展,加入了一个新的维度——上下文信息。从基于上下文融入推荐生成过程具体阶段的角度划分,可以将上下文感知推荐生成技术分为如下3种:上下文预过滤(contextual pre-filtering)、上下文后过滤(contextual post-filtering)、上下文建模(contextual modeling)。上下文预过滤是指利用当前上下文信息将无关的用户偏好数据过滤掉,构建一个只与当前上下文信息相关的数据集;然后通过利用传统推荐技术(基于内容的过滤、协同过滤、基于知识的过滤、混合式过滤等)处理这

到稿日期:2016-01-19 返修日期:2016-05-24 本文受国家自然科学基金项目(60803086),国家科技支撑计划子课题(2013BAH21B02-01),北京市自然科学基金项目(4153058,4113076)资助。

何 明(1975-),男,博士,副教授,主要研究方向为推荐系统、数据挖掘、机器学习,E-mail:heming@bjut.edu.cn;刘 毅(1989-),男,硕士生,主要研究方向为推荐系统、数据挖掘;常盟盟(1987-),男,硕士生,主要研究方向为机器学习;吴小飞(1991-),男,硕士生,主要研究方向为信息检索。

些过滤后的数据来进行潜在偏好预测，生成满足当前上下文约束的推荐结果。上下文后过滤是指首先不考虑上下文因素，利用传统二维推荐技术处理不含上下文信息的推荐数据来预测用户偏好，然后根据当前上下文信息筛选掉不相关的推荐结果或者调整 Top-N 排序列表。上下文建模不会首先忽略掉上下文信息对用户偏好的影响，而是将上下文信息融入推荐的整个过程，通过设计合适的算法和模型来处理多维度上下文信息，以此来预测用户的潜在偏好。

本文提出了一种基于项目评分分裂(Item Splitting)的上下文感知推荐方法(Context-Aware Splitting Approach，CASA)。首先在多维度上下文信息下依据不同分裂标准对项目进行分裂，然后对分裂结果按照上下文维度进行聚类，最后融合协同过滤算法进行潜在偏好预测。同时在 LDOS-CoMoDa 数据集上进行对比实验，与其他上下文偏好提取技术进行比较，所提算法的实验效果明显优于其他算法。

本文第 2 节具体介绍了基于项目评分分裂方法的分裂标准、分裂过程的实现，以及依据分裂方法的聚类实现；第 3 节则基于 SVD 构建推荐引擎；第 4 节分析了实验结果，对比了不同推荐算法在不同分裂标准上的效果，同时也比较了不同的上下文感知推荐算法之间的推荐精度；最后对全文进行了总结。

2 上下文项目评分分裂方法

2.1 问题建模

传统的协同过滤算法通过将用户对项目的评分数据进行计算来实现推荐。用户的评分数据 r 可以用一个 m×n 阶的用户-项目评分矩阵 R 表示。m 行代表 m 个用户，n 列代表 n 个项目，第 m 行第 n 列的元素 r_{mn} 代表用户 m 对项目 n 的评分。对于给定的一组评分{?,1,2,3,4,5}，协同过滤算法就是通过已有的评分{1,2,3,4,5}来预测缺失的评分“?”。

本文将协同过滤“用户-项目”二维评分效用模型 $R:Users\times Items->Rating$ 扩展为包含上下文信息的多维评分效用模型 $R:Users\times Contexts\times Items->Rating$[3]。上下文信息可以表示为一组特定属性的集合(如时间、地理位置、人员等)：$C=\{C_1,C_2,\cdots,C_k\}$，$C_j=\{c_1,c_2,\cdots\}$，$C_j\in C$。其中，C_1，C_2，…，C_k 代表上下文维度，c_1，c_2，…代表上下文条件，它是上下文维度中的特定值。例如，C_1：Weather＝{c_1：sunny，c_2：cloudy，c_3：raining，c_4：snowing}，C_2：Time＝{c_1：weekday，c_2：weekend}，C_3：Companion＝{c_1：friend，c_2：girlfriend，c_3：family}。将用户对项目的评分从传统的二维评分模型扩展为包含多种上下文信息的多维评分效用模型，即用户 u 对项目 i 的评分 r_{ui} 与一组上下文信息 $c(u,i)=(c_1,\cdots,c_j,\cdots)$($c_j\in C_j$)相关。一个与上下文相关的电影评分如表 1 所列。

表 1 上下文相关的项目(电影)评分

User	Item	C_1(Time)	C_2(Location)	C_3(Companion)	Rating
u_1	i_1	Weekend	Home	Friend	3
u_1	i_1	Weekend	Cinema	Girlfriend	5
u_1	i_1	Weekday	Home	Family	?

对于 m 个用户，n 个项目，k 个上下文信息，可将其分别表示为用户集 $U=\{u_1,u_2,\cdots,u_m\}$，项目集 $I=\{i_1,i_2,\cdots,i_n\}$和上下文集 $C=\{C_1,C_2,\cdots,C_k\}$。对于每个用户 u，在不同的上下文维度 C_j($C_j\in C$)下有相对应的项目评分向量 $r_u=\{r_{u1},r_{u2},\cdots,r_{un}\}$。

2.2 项目分裂原理

项目评分分裂方法的基本思想是从上下文维度 C_j($j\in[1,k]$)中发现用于分裂项目 i 的上下文条件 $c\in C_j$。经过分裂后，i 的评分向量 $r_i=(r_{1i},\cdots,r_m)$分裂为两个子向量 r_{i_c}($c_j=c$)和 $r_{i_{\bar{c}}}$。对于每一个子向量，分别定义参与集 PSr_{i_c} 和非参与集 $PSr_{i_{\bar{c}}}$，即

$$PSr_{i_c}=\{r_{i_c}\mid r_{ui}=r_{ui},c(u,i)=c,u\in U,i\in I\}$$

$$PSr_{i_{\bar{c}}}=\{r_{i_{\bar{c}}}\mid r_{ui_{\bar{c}}}=r_{ui},c(u,i)\neq c,u\in U,i\in I\}$$

对于最优分裂 c 的选择，基于这样的观察：当 PSr_{i_c} 和 $PSr_{i_{\bar{c}}}$ 之间的差异性指数越大，表明 r_i 受 c 的影响越大，对 c 的依赖性越强；相反，则表明 c 对 r_i 的影响较小，r_i 对 c 的依赖性越弱。图 1 描述了 i 在 $c\in C_j$ 上的分裂过程。通常情况下，可能会有更多的项目需要进行分裂。m 个用户、n 个项目的 $m\times n$ 阶用户-项目评分矩阵经过分裂后得到 $m\times(n+l)$ 阶矩阵，l 表示分裂的项目数。原始矩阵经过分裂后，矩阵中评分的数量保持不变，丢弃了原有的 l 个项目，同时生成了 $2l$ 个新项目。

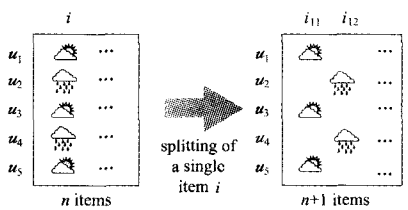

图 1 项目 i 的分裂过程

2.3 项目分裂标准和聚类算法

2.3.1 上下文信息系统

基于包含上下文信息的多维评分效用模型 $R:Users\times Context\times Items->Rating$，定义上下文信息系统为 $CIS=\langle U,R,V,f\rangle$，其中 U 是对象的集合；$R=C\cup D$ 是属性集合，$C=\{C_1,C_2,\cdots,C_k\}$是上下文属性集，$C_j=\{c_1,c_2,\cdots\}$，$C_j\in C$；D 是评分属性集，可以按照评分 $Rating$ 将对象分类，即项目的评分类；$V=\bigcup_{r\in R}V_r$，V_r 是属性值的集合，V_r 表示属性 $r\in R$ 的属性值范围，即属性 r 的值域；$f:U\times R\to V$ 是一个信息函数，它指定 U 中的每一个对象 x 的属性值，即对 $\forall x\in U$，$\forall r\in R$，$f(x,r)\in V_r$。

2.3.2 分裂标准

基于 2.2 节中项目分裂的原理和统计学方法，通过计算项目 i 在上下文条件 c 下 PSr_{i_c} 和 $PSr_{i_{\bar{c}}}$ 之间的差异性指数为项目 i 的分裂提供依据。本文采用了 t 系列假设检验标准，具体包括以下标准。

$t_{mean}(i,c)$：双样本 t 检验。计算项目 i 在上下文条件 c 下分裂后形成的两个项目 i_c 和 $i_{\bar{c}}$ 评分子集平均值的差。t 值越大表明两个评分子集间的差异性越大。t_{mean} 定义为：

$$t_{mean}=\left|\frac{\mu_{i_c}-\mu_{i_{\bar{c}}}}{\sqrt{s_{i_c}/n_{i_c}+s_{i_{\bar{c}}}/n_{i_{\bar{c}}}}}\right| \tag{1}$$

其中，μ_i 表示项目 i 评分的平均值，s_i 表示项目 i 评分的方

差,n_i 表示项目 i 评分的个数。

$t_{prop}(i,c)$:双比例 z 检验。计算项目 i 在上下文条件 c 下分裂后形成的两个项目i_c和$i_{\bar{c}}$评分子集中高评分和低评分所占比例的差。$t_{prop}(i,c)$定义为:

$$t_{prop}(i,c)=\frac{p_{i_c}-p_{i_{\bar{c}}}}{\sqrt{p(1-p)(1/n_{i_c}+1/n_{i_{\bar{c}}})}} \tag{2}$$

其中,$p=\frac{p_{i_c}n_{i_c}+p_{i_{\bar{c}}}n_{i_{\bar{c}}}}{n_{i_c}n_{i_{\bar{c}}}}$,$p_{i_c}$($p_{i_{\bar{c}}}$)是项目 i_c($i_{\bar{c}}$)中的高(低)评分的比例,n_{i_c}($n_{i_{\bar{c}}}$)是项目 i_c($i_{\bar{c}}$)的数目。

$t_{IG}(i,c)$:计算上下文条件 c 对项目 i 的信息增益(Information Gain,IG)。$t_{IG}(i,c)$定义为:

$$t_{IG}=H(i)-H(i_c)P_{i_c}+H(i_{\bar{c}})P_{i_{\bar{c}}} \tag{3}$$

其中,$H(i)$($H(i_{\bar{c}})$)是项目 i 评分分布的香农熵,P_{i_c}($P_{i_{\bar{c}}}$)是i_c($i_{\bar{c}}$)在项目 i 中所占的比例。

$t_{chi}(i,c)$:卡方检验。计算项目 i 在上下文条件 c 下分裂后形成的两个项目i_c和$i_{\bar{c}}$中高评分和低评分的比例差。

2.3.3　单维上下文评分矩阵的划分

按照上下文维度 $C=\{C_1,C_2,\cdots,C_k\}$($C_j=\{c_1,c_2,\cdots\}$,$C_j\in C$),可以将评分矩阵 R 划分为 k 个子矩阵 $K_1\ K_2\cdots K_k$。每个 K_i 都是在相同单维度上下文 C_j 上进行分裂的用户-项目评分子矩阵。例如,C_1:Time=$\{c_1$:weekday,c_2:weekend$\}$,C_2:Location=$\{c_1$:home,c_2:cinema$\}$,C_3:Companion=$\{c_1$:friend,c_2:girlfriend,c_3:family$\}$,可以将表 2 所列的包含 C_1,C_2 和 C_3 的三维上下文信息的评分矩阵根据项目分裂时的上下文维度划分成 3 个分别包含单维上下文信息的评分矩阵:C_1-Metric,C_2-Metric 和 C_3-Metric,如表 3—表 5 所列。这样,就将包含多维上下文信息的多维评分效用模型转换为单维度上下文评分效用模型,即用户 u 对项目 i 的评分 r_{ui} 仅依赖于单维上下文信息。本文基于单维上下文的划分,可以把多维推荐空间问题降维到标准的二维用户-项目空间,从而允许使用任何传统的推荐评分预测技术。

表 2　上下文相关项目(电影)评分

User	Item	C_1(Time)	C_2(Location)	C_3(Companion)	Rating
u_1	i_1	weekend	home	friend	3
u_2	i_2	weekday	cinema	girlfriend	5
u_3	i_3	weekday	home	family	1
u_4	i_4	weekend	cinema	girlfriend	4
u_5	i_5	weekend	home	friend	5
u_6	i_6	weekday	cinema	family	2

表 3　C_1-Metric 评分矩阵

	i_1	i_2
u_1	3	
u_2		5

表 4　C_2-Metric 评分矩阵

	i_3	i_4
u_3	1	
u_4		4

表 5　C_3-Metric 评分矩阵

	i_5	i_6
u_5	5	
u_6		2

2.4　项目算法和矩阵划分算法

算法 1　项目分裂算法

Input:CIS=〈U,REDC∪D,V,f〉,threshold θ

Output:PSr_{i_c} and $PSr_{i_{\bar{c}}}$ of all splitting items with respect to contextual condition c

1. set initialized $PSr_{i_c}=PSr_{i_{\bar{c}}}=\emptyset$
2. for eachitem i in U do
   2.1. for $\forall C_j\in$(REDC) do
      2.1.1. for $\forall c\in C_j$ do
      2.1.2. generated after split r_{i_c} ($c_j=c$)and $r_{i_{\bar{c}}}$ ($c_j\neq c$).
      2.1.3. calculate significant difference between r_{i_c} and $r_{i_{\bar{c}}}$ VD(r_{i_c}, $r_{i_{\bar{c}}}$).
      2.1.4. end for
   2.2. end for
3. set $c_{max}\leftarrow\arg\max_c(VD(r_{i_c},r_{i_{\bar{c}}}))$.
4. if $VD(r_{i_{c_{max}}},r_{i_{\bar{c}_{max}}})$ then
   4.1. add split item i to PSr_{i_c} and $PSr_{i_{\bar{c}}}$ respectively go to 5.
   4.2. else
   4.3. item i left unchanged,go to 2.
   4.4. end if
5. output PSr_{i_c} and $PSr_{i_{\bar{c}}}$.
6. end for
7. end

对于所有项目,算法 1 选择最优的一个上下文条件 c_{max} 作为该项目分裂的依据。由于多维上下文条件组合形成的分裂会进一步导致严重的评分数据的稀疏性,同时,模型复杂度的增加也会导致过拟合,因此本文采用基于单维上下文条件的分裂方法。将最优上下文条件 c_{max} 作为项目分裂的依据,在不太影响推荐性能的同时,也可以在一定程度上缓解评分稀疏性问题。在时间复杂度方面,给定 n 个项目、m 个用户、k 个上下文维度,每个上下文维度有 d 个不同的上下文条件,算法 1 的时间复杂度为 $O(nmkd)$。

算法 2　聚类算法

Input:PSr_{i_c} and $PSr_{i_{\bar{c}}}$ of all splitting items with respect to contextual condition c

Output:all splitting items classification with respect to contextual condition C

1. get contextual condition c from PSr_{i_c}
2. for each item i in PSr_{i_c} do
   2.1. for $\forall C\in C_j$ do
   2.2. generated matric with the condition C.
   2.3. end for
   2.4. end for
3. output matric with single condition C.
4. end

本文将 2.3.3 节所述的单维上下文评分矩阵的划分作为聚类过程来处理。对于所有经过分裂的项目,算法 2 进一步将每个项目 i 按照分裂时参照的上下文维度进行聚类。对于包含 k 个上下文维度和 n 个项目的分裂,算法 2 的时间复杂度为 $O(nk)$。

在多维上下文信息中,算法 1 根据分裂标准进行判别,依据是否产生分裂来确定在哪个上下文维度下用户 u 对项目 i 的评分是最高的,选出影响最大的上下文维度。算法 2 按照算法 1 分裂时所对应的上下文维度将用户-项目对进行聚类,从而将包含上下文信息的多维推荐空间降维到标准的二维用户-项目空间,以便为进一步采用传统的推荐算法提供必要条件。

3 基于协同过滤的推荐过程

3.1 奇异值分解 SVD

奇异值分解(Singular Value Decomposition,SVD)作为一种常见的矩阵分解方法,在信息检索中利用 SVD 的方法包括隐性语义分析(Latent Semantic Analysis,LSA)或者隐性语义索引(Latent Semantic Indexing,LSI),推荐系统中首先运用 SVD 从数据中构建一个主题空间,然后在该空间下计算相似度。其具有简化数据、去除噪声、优化算法结果的优点。经过 2.3.3 节单维上下文评分矩阵划分成的新矩阵会包含很多噪声和冗余数据,利用 SVD 的实现,从噪声数据中抽取相关特征,从而达到去除噪声和冗余信息的目的。

SVD 奇异值分解将一个复杂的矩阵转化成几个小矩阵相乘的形式,每个小矩阵表示原矩阵的一些重要的特征。对于一个 $m\times n$ 的数据集矩阵 $Data$,SVD 将其分解成 3 个矩阵 U,Σ 和 V^{T},其形式如下:

$$Data_{m\times n}=U_{m\times m}\Sigma_{m\times n}V^{\mathrm{T}}_{n\times n} \tag{4}$$

其中,左奇异向量 U 是一个 $m\times m$ 的矩阵;Σ 是分解中构建的一个 m×n 的矩阵,该矩阵只有对角元素,其余元素都为 0,对角线上的元素称为奇异值(*Singular Value*),它们对应了原始数据集矩阵 $Data$ 的奇异值(矩阵 $Data * Data^{\mathrm{T}}$ 的特征值的平方根),其值是按从大到小排列的,且迅速下降;右奇异向量 V^{T} 是一个 $n\times n$ 矩阵。矩阵 $Data$ 的 SVD 分解示意图如图 2 所示。

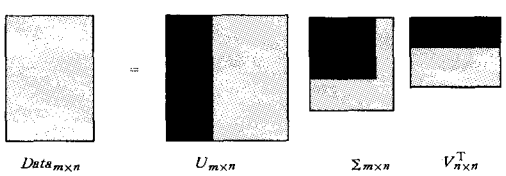

图 2 矩阵 *Data* 的 *SVD* 分解示意图

图 2 中浅灰色区域是原始数据,黑色区域是矩阵近似计算仅需要的数据。

通常在某个奇异值的数目(R 个)之后,其他的奇异值都是 0,这就表示当前数据集中仅有 R(R 是一个远小于 M 和 N 的数)个重要特征,其余的都是冗余特征或者噪声信息。在具体计算当中,能有很多启发式的策略能确定要保留的奇异值的数目,其中一种方法就是保留矩阵中 90% 的能量信息,为了得到总能量信息,求其所有奇异值的平方和,然后累加奇异值之和直到总值的 90% 为止;另外一种启发式策略就是当奇异值的数目达到一定的数量级(万级)时,可以保留前面的 2000 或者 3000 个,这种策略在准确度上有一定的误差,因为我们无法确定其前 3000 个奇异值就包含了 90% 的能量信息,但是在实际操作中更容易实施。

3.2 基于协同过滤的推荐引擎

矩阵分解技术被认为是传统推荐系统领域中最有效的手段之一,但是它只能处理二维推荐数据,不适用于多维上下文数据的推荐。本文通过项目评分分裂方法将多维推荐空间问题降维到标准的二维用户-项目空间(该空间包含了单一维度的上下文信息),再结合传统的矩阵分解技术以及协同过滤算法构建推荐引擎。

3.2.1 相似度计算方法

计算用户间相似性的方法有很多种,这里介绍余弦相似度(cosine similarity)和相关相似度(correlation similarity)[6]。

余弦相似度:用户评分被看作是 n 维项目空间上的向量,用户间的相似度通过向量间的余弦夹角 θ 度量,如果夹角为 90°,则相似度为 0;如果两个向量的方向相同,则相似度为 1.0。设用户 A 和用户 B 在 n 维项目空间上的评分分别表示为向量 $\vec{A}$ 和向量 $\vec{B}$,则用户 A 和用户 B 之间的相似度 $sim(A,B)$ 为:

$$sim(A,B)=\cos\theta=\frac{\vec{A}\times\vec{B}}{\|\vec{A}\|\ \|\vec{B}\|} \tag{5}$$

相关相似度:通过 Pearson 相关系数度量,用户 A 和用户 B 之间的相似度 $sim(A,B)$ 为:

$$sim(A,B)=\frac{\sum_{i\in I_{AB}}(R_{A,i}-\bar{R}_A)(R_{B,i}-\bar{R}_B)}{\sqrt{\sum_{i\in I_A}(R_{A,i}-\bar{R}_A)^2}\sqrt{\sum_{i\in I_B}(R_{B,i}-\bar{R}_B)^2}} \tag{6}$$

其中,I_{AB} 表示用户 A 和用户 B 共同评分的项目集合,$R_{A,i}$ 表示用户 A 对项目 i 的评分,$\bar{R}_A$($\bar{R}_B$)表示用户 A(B)对项目的评分的平均值。

3.2.2 预测方法

本实验采用一种根据不同用户对相同项目的已有评分来预测未评分项目估计评分值的方法。其实现方式有很多种,其中 standEst 和 svdEst 是两种常用的预测方式。这里给出 standEst 的基本运行过程:

1)得到数据集中的项目数量,并对用于计算估计评分值的变量进行初始化。

2)对用户评过分的每个物品进行遍历,并将它和其他项目进行比较,寻找它们之间的重合元素。若评分值为 0(用户对该项目没有评分),则跳过这个项目;如果两者没有任何重合的元素,则相似度为 0,跳过该项目,并继续寻找下一个项目;如果存在重合的项目,则基于这些重合项目计算相似度。

3)相似度会不断累加,每次计算时还需要考虑相似度和当前用户评分的乘积。

4)对相似度评分的乘积进行归一化。通过除以所有评分的总和,使得最后的评分值在 0~5 之间,在推荐最后阶段利用这些评分值进行排序,从而得出预测值。

svdEst 与 standEst 过程相似,其差别是在步骤 2)中 svdEst 对数据进行了 SVD 分解,分解之后只利用包含了 90%能量的奇异值构建一个对角矩阵,利用 U 矩阵将项目转换到低维空间。本文的实验就是采用的这种方法。

3.2.3 基于项目相似度的推荐引擎

基于项目(用户)的相似度计算就是利用相似度方法计算两个项目(用户)之间的距离。基于项目(用户)的时间复杂度会随着项目(用户)数量的增加而增加,而根据本文实验所用到的数据集的特性,项目增加的数量会远远低于用户增加的数量,因此本文选择基于项目的相似度构建推荐引擎。

利用 3.2.1 节所提到的相似度计算方法和 3.2.2 节的预测方法来构建基于项目相似度的推荐引擎。对于用户 u(需

要被推荐项目的用户）所有未评分的项目，应用评估策略得到预测值，从而取得 TOP-N 的推荐结果，即得到排名靠前的 N 个项目（最有可能被推荐给用户的项目），这里取 TOP-1 项目作为用户 u 的推荐项目。通常 N 值需要提前设定，如果不指定 N 的大小，则需要设定默认值。其构建过程为：

1）选取合适的相似度计算方法和评估策略；

2）对给定的用户建立一个未评分的项目列表；

3）在所有的未评分项目上进行循环计算，对于每个未评分的项目，通过评估策略来产生该物品的预测评分；

4）所有的预测评分会以项目的编号和估计得分值保存；

5）按照估计得分对该列表进行排序（从大到小），因此第一个值就是最大值。

3.3　推荐过程

本文利用基于项目评分的分裂方法对包含多维上下文信息的数据集进行数据处理，形成传统的二维矩阵，然后利用 SVD 进行推荐。对于用户 u，要为其推荐合适的项目 i，即选取用户 u 对未评分项目预测评分最大的项目的过程为：

1）对包含多维度上下文信息的数据集，根据分裂标准进行基于项目的分裂。

2）将数据集在单维度上下文 C_j 上分裂后的所有项目聚类，形成多个用户-项目评分子矩阵。例如，用户 u_1 对项目 i_1、用户 u_2 对项目 i_4、用户 u_8 对项目 i_6 都在上下文 C_j 上进行分裂，即不同用户对不同项目在相同的上下文进行了分裂，则认为上下文 C_j 是对用户-项目影响最大的因素。因此，将所有在该上下文上 C_j 上分裂的项目聚类成用户-项目评分子矩阵 C_j-Metric。

3）对不同上下文维度形成的子矩阵 C_j-Metric 中，利用相似度计算方法和评分预测算法计算用户 u 对单个项目的评分。

4）根据推荐引擎计算用户 u 对当前矩阵 C_j-Metric 所有项目的估计评分值，对产生的估值最高的 N 个推荐结果进行排序，最终得到估值最高的那个即为最终推荐结果。

4　实验分析

4.1　实验数据集

LDOS-CoMoDa 数据集[7]是一个面向电影推荐领域的上下文感知推荐的真实数据集，通过问卷调查的形式，所有评分均是通过用户观影后及时记录当时的观影环境得到的，而不是通过假设观影环境或者在用户观看电影一段时间后依靠回忆得到的，在一定意义上反映了对客观事实的评价效果。数据集包含了同一用户对同一电影的不同评分，考虑到上下文因素的不同，如果同一用户多次观看了同一部电影，并在观影后给出了不同的评分，仍然认为每次的评分都是有效的。

LDOS-CoMoDa 数据集采集了 113 位用户对 1186 部电影的 2094 条评分数据，这些评分数据同时包含了 12 种上下文因素（表 6 列出了这 12 种上下文因素的属性），评分范围从 1（讨厌）～5（喜欢）。为了将数据集应用于分裂标准中，图 3 给出了 LDOS-CoMoDa 数据集中高（大于 3）低评分的分布比例。其数据摘要如表 7 所列。

表 6　LDOS-CoMoDa 数据集上 12 种上下文因素的基本属性

变量名	Rg	MVR	描述
time	4	0.017	morning, afternoon, evening, night
daytype	3	0.015	working day, weekend, holiday
season	4	0.017	spring, summer, autumn, winter
location	3	0.016	home, public place, friend's house
weather	5	0.021	sunny/clear, rainy, stormy, snowy, cloudy
social	7	0.013	alone, partner, friends, colleagues, parents
endEmo	7	0	sad, happy, scared, surprised, angry
dominantEmo	7	0	sad, happy, scared, surprised, angry
mood	3	0	positive, neutral, negative
physical	2	0.022	healthy, ill
decision	2	0.021	user's choice, given by other
interaction	2	0.020	first, n-th

注：Rg 表示变量的分类数目，MVR 表示缺失的评分值。

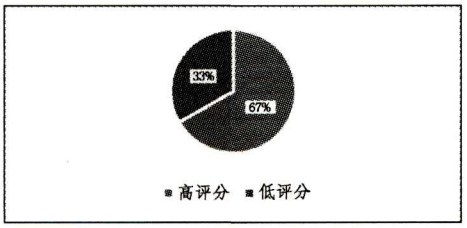

图 3　LDOS-CoMoDa 数据集评分分布

表 7　LDOS-CoMoDa 数据集

用户数	电影数	评分条数	上下文个数	评分范围	稀疏度
113	1186	2094	12	1～5	1.6%

4.2　评价标准及实验参数配置

4.2.1　实验目的

本实验的目的主要包括以下几个方面：

1）基于本文提出的上下文项目评分分裂方法，找出不同的分裂标准（2.3.2 节所列分裂标准）中最优的分裂标准；

2）将项目评分分裂方法与传统的基于用户的协同过滤（User-based Collaborative Filtering，UBCF）、基于项目的协同过滤（Item-based Collaborative Filtering，IBCF）和矩阵分解（MF）相结合，并对推荐效果进行比较；

3）比较基于上下文项目评分分裂方法和其他上下文感知推荐算法的推荐精度，验证基于上下文项目评分分裂的方法能有效提升推荐精度；

4）受 Yong Zheng 等人[8]的启发，构建一种新的评估方法 CPrecision（ContextualPrecision）和 CROC（Contextual ROC），验证基于上下文项目评分分裂方法是一种处理含有上下文信息的有效预过滤处理方法。

4.2.2　评价标准

评价推荐系统推荐质量的度量标准主要有统计精度度量方法（Prediction Error）（比如 MAE，RMSE）、决策支持精度度量方法（IR metrics）（比如 Precision，Recall，ROC）和排名度量方法（RankingMetrics）（比如 MAP，NDCG）。

统计精度度量方法中的均方根偏差（Root Mean Squared Error，RMSE）[9]的定义如下：

$$RMSE=\sqrt{\frac{\sum_{i=1}^{N}(r_i-\bar{r}_i)}{N}} \tag{7}$$

其中，N 是用户对项目评分的总个数，r_i 是用户对项目 i 的真

实评分，$\bar{r}_i$ 是用户对项目 i 的预测评分。

精度(precision)定义为预测命中项目评分数与总的预测项目评分数的比值。查全率(recall)定义为正确预测的项目数与总的用户请求项目数的比值。对于上下文推荐系统，将上下文信息添加到这两种评测标准中从而形成可以评测含有上下文信息的评分新标准，将其称为 CPrecision 和 CRecall。CPricsion 表示在单维度上下文信息中正确预测项目评分数与在当前上下文中总的项目数评分的比值。同样，CRecall 表示在单维度上下文信息中预测命中的数量与用户总的偏好选项的比值。

ROC 曲线是度量信息过滤系统能力的曲线，同时也是对同一信号刺激反应的呈现，反应越大则说明所使用算法的效果越强烈。ROC 曲线是根据一系列不同的二分类方式(阈值或决定阈)，以真阳性率(truepositive rate)为纵坐标、假阳性率(false positive rate)为横坐标绘制的曲线。ROC 曲线将真、假阳性率以图示的方式结合在一起，曲线下的面积越大，判断价值越高。真阳性率指实际为真值的判断作为真值的概率。假阳性率指实际为假值的判断作为假值的概率。在本文中将其分为经过预过滤处理的项目数与用户偏好相关项目两类，然后进行统计分析。将上下文信息加入到 ROC 曲线当中，形成 CROC 曲线，结合上文提到 CPrecision 和 CRecall 方法来评价包含上下文信息的信息过滤系统的过滤能力。

4.2.3 环境及参数配置

本实验采用了 4 种分裂标准即 t_{mean}，t_{chi}，t_{prop} 和 t_{IG} 来评估推荐算法的性能。各标准所对应的参数配置如表 8 所列。其中，R 为评分参数，$\in$ 为预定义的项目分裂阈值。实验代码由 Python 实现。实验代码运行于 PC 机，PC 机的配置为 Intel (R)Core(TM) i5-4590 CPU @ 3.30GHz，8GB 内存，操作系统为 Windows 7，Python 版本为 2.7.10。

表 8 参数配置

分裂标准	参数配置	说明
t_{mean}	$\in=0.05$	$t_{mean}\geqslant\in$ 的项目进行分裂
t_{chi}	$R=3,\in=0.05$	评分$>R$ 的项目为高评分项目；评分$\leqslant R$ 的项目为低评分项目
t_{prop}	$R=3,\in=0.05$	评分$>R$ 的项目为高评分项目；评分$\leqslant R$ 的项目为低评分项目
t_{IG}	$\in=0.20$	$t_{IG}\geqslant\in$ 的项目进行分裂

对于实验参数 $\in$ 和 R 的取值，本文参照了文献[3]。LDOS-CoMoDa 数据集中所有的评分在 1～5 之间，选取中间值 $R=3$ 作为项目高低评分的标准。

4.3 推荐效果比较

为了验证本文提出的方法的有效性，引入了 3 种传统的推荐算法：UBCF，IBCF 和 MF，并比较它们的推荐效果。Koren 提出了非对称 SVD(Asymmetric SVD，AsySVD)，BiasMF(MF with rating bias)和SVD++3 种基于矩阵分解的算法[10]，并对它们的推荐效果进行了比较，发现 3 种算法的差异并不明显。本文选用了非对称 SVD 作为推荐引擎。

采用开源推荐引擎 MyMediaLitev3.07[11] 中 k-近邻算法(邻域大小设为 30)来评估基于用户的协同过滤和基于项目的协同过滤；对于矩阵分解的评估，则采用 Python 基础库 NumPy v1.9.3 构建的推荐引擎(3.2.3 节基于项目相似度的推荐引擎)。

比较基于项目评分分裂方法基于上下文感知矩阵分解(Context-Aware Matrix Factorization，CAMF)以及基于差分模型[12](Differential Context Modeling，DCM)的算法的推荐精度。文献[13-14]对传统矩阵分解进行拓展以用于上下文偏好的提取，从而将上下文信息融入推荐过程；文献[15-17]通过建立机器学习模型来提取上下文偏好；本文则提出一种基于项目分裂的方法进行上下文偏好的提取。实验中，选择 CAMF_C[13]，CAMF_CI[13]，CAMF_CU[14]，DCR[15-16] 和 DCW[17] 作为对比参照。

4.3.1 分裂标准间的比较

为了找出分裂标准中的最优标准，对比了 UBCF，IBCF，SVD 3 种推荐算法在 t_{mean}，t_{chi}，t_{prop} 和 t_{IG} 4 种分裂标准下的 RMSE，如表 9 所列。表 9 中还同时包含了 SP 和 $Sparsity$ 两个比较标准：SP 表示分裂项的百分比(即分裂项目数/项目总数)，$Sparsity$ 表示经过项目分裂后评分矩阵的稀疏度。

表 9 不同分裂标准下的 RMSE 比较

算法	t_{mean}	t_{chi}	t_{prop}	t_{IG}
UBCF	1.040	1.021	1.028	1.043
IBCF	1.030	1.024	1.026	1.034
SVD	1.020	1.011	1.016	1.020
SP	3.0%	6.0%	3.8%	3.5%
Sparsity	98.5%	98.5%	98.5%	98.5%

在同一数据集下，不同的分裂标准可以通过 $RMSE$，SP 和 $Sparsity$ 值进行评价。$RMSE$ 值越小表示推荐越精确；SP 值越大说明分裂项目数就越多，上下文偏好的提取就越精确；$Sparsity$ 值越小说明稀疏性就越小。表 9 中下划线标示的数值表示在 LDOS-CoMoDa 数据集中的最优结果。由表 9 可知，t_{chi} 分裂标准在 UBCF，IBCF，SVD 3 种算法中的 $RMSE$ 值都最小，SP 值最大，即在相同数据集下的实验结果表明，与其他 3 种分裂标准相比，t_{chi} 分裂标准在 LDOS-CoMoDa 数据集上的表现是最优的。

实验中，基于 UBCF，IBCF 和 SVD 构建了 3 种推荐引擎。表 9 同时也给出了 3 种算法在 t_{mean}，t_{chi}，t_{prop} 和 t_{IG} 4 种分裂标准下的 $RMSE$ 值。从表 9 中可以看到，在同一分裂标准下，SVD 的 $RMSE$ 值都是最小的，即推荐精度最高。而基于 UBCF，IBCF 构建的推荐引擎在不同的分裂标准下的推荐精度各有不同。实验结果表明，本文采用的基于矩阵分解的 SVD 可以有效缓解 LDOS-CoMoDa 数据集的数据稀疏性问题，同时具有较高的推荐精度。

由表 9 可知，在 LDOS-CoMoDa 数据集上，采用 t_{chi} 分裂标准并基于 SVD 构建的推荐引擎(即本文所采用的方法)会明显提升推荐精度。

4.3.2 上下文感知推荐算法间的比较

为了验证基于上下文项目评分分裂方法是否有效提升了推荐精度，对比了 CAMF 和 DCM，并选取 CAMF_C，CAMF_CI，CAMF_CU，DCR 和 DCW 作为参照。图 4 示出了这 6 种预过滤算法的 $RMSE$ 值对比。从图 4 的实验结果比较中可

以明显看出，本文提出的基于项目评分分裂的方法较其他算法具有较低的 *RMSE* 值，表明其推荐精度较高。图 5 和图 6 分别给出了基于 CPrecision 和 CROC 曲线测试的结果，实验中将大于评分阈值的项目作为参与 CPrecision 计算相关项。从图 5 中可以看出，相比于上下文感知预过滤算法 CAMF_C，CAMF_CI，CAMF_CU，DCR 和 DCW，项目评分分裂方法的推荐精度高于其他算法的推荐精度。CROC 曲线作为一个很重要的辅助指标，图 6 中选取 CRecall 为真阳性率（纵坐标），fallout 为假阳性率（横坐标），通过两者之间的面积来评价实验结果。由图 6 中可以看出，项目评分分裂方法对同一信号的刺激反应更加强烈，即项目评分分裂方法具有较高的推荐精度。

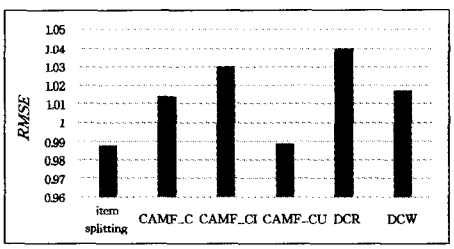

图 4　不同上下文感知算法的 *RMSE*

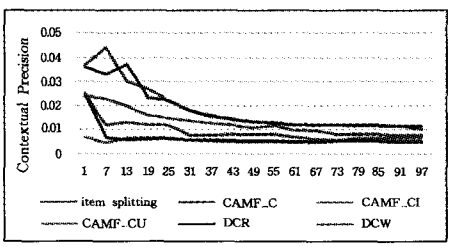

图 5　不同上下文感知算法的 CPrecision 比较

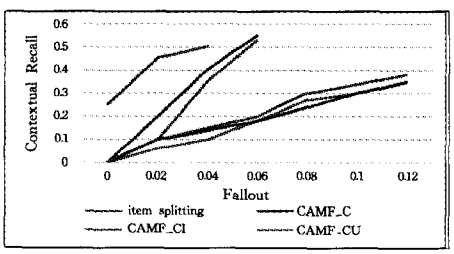

图 6　不同上下文感知算法的 CROC 曲线图

结束语　为了提高上下文感知推荐的精度，提出了一种基于项目评分分裂的上下文预过滤处理方法，将包含多维上下文信息的多维评分效用模型转换为单维度上下文评分效用模型，然后结合传统推荐算法进行推荐预测。实验表明该方法是一种有效的处理包含上下文信息数据的方法。在今后的工作中，我们将进一步研究基于用户的分裂以及基于项目和用户的混合分裂方法，以期进一步提升推荐质量。

参 考 文 献

[1] ZENG C, XING C X, ZHOU L Z. A Survey of Personalization Technology [J]. Journal of Software, 2002, 13(10): 1952-1961.

[2] ADOMAVICIUS G, SANKARANARAYANAN R, SEN S, et al. Incorporating contextual information in recommender systems using a multidimensional approach [J]. ACM Transactions on Information Systems, 2005, 23(1): 103-145.

[3] ADOMAVICIUS G. Incorporating contextual information in recommender systems using a multidimensional approach [J]. ACM Transactions on Information Systems, 2005, 23(1): 103-145.

[4] ADOMAVICIUS G, RICCI F. RecSys'09 workshop 3: workshop on context-aware recommender systems (CARS-2009) [C]// Proceedings of the 2009 ACM Conference on Recommender Systems (RecSys 2009). New York, NY, USA, 2009: 423-424.

[5] DEY A K. Understanding and using context [J]. Personal Ubiquitous Computer, 2001, 5(1): 4-7.

[6] DENG A L, ZHU Y Y, SHI B L. A Collaborative Filtering Recommendation Algorithm Based on Item Rating Prediction [J]. Journal of Software, 2003, 14(9): 1621-1628.

[7] KOŠIR A, ODI A, KUNAVER M, et al. Database for contextual personalization [J]. ElektrotehniskiVestnik/electrotechnical Review, 2011, 78(5): 270-274.

[8] ZHENG R B, MOBASHER B. Splitting approaches for context-aware recommendation: an empirical study [C]// Proceedings of the 29th Annual ACM Symposium on Applied Computing. 2014: 274-279.

[9] SARWAR B, KARYPIS G, KONSTAN J, et al. Item-Based collaborative filtering recommendation algorithms [C]// Proceedings of the 10th International World Wide Web Conference. 2001: 285-295.

[10] KOREN Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model [C]// Proceedings of ACM Conference on Knowledge Discovery and Data Mining (KDD). 2008: 426-434.

[11] Gantner Z, Rendle S, Freudenthaler C, et al. Mymedialite: A free recommender system library [C]// Proceedings of ACM Conference on Recommender Systems. 2011: 305-308.

[12] ZHENG Y, BURKE R, MOBASHER B. Differential context modeling in collaborative filtering [C]// Proceedings of School of Computing Research Symposium. 2013.

[13] BALTRUNAS L, LUDWIG B, RICCI F. Matrix factorizationtechniques for context aware recommendation [C]// ACM Conference on Recommender Systems. 2011: 301-304.

[14] ODIC A, TKALCIC M, TASIC J F, et al. Relevant context in a movie recommender system: Users' opinion vs. statistical detection [C]// ACM RecSys. 2012.

[15] ZHENG Y, BURKE R, MOBASHER B. Differential context relaxation for context-aware travel recommendation [C]// 13th International Conference on Electronic Commerce and Web Technologies. 2012: 88-99.

[16] ZHENG Y, BURKE R, MOBASHER B. Optimal feature selection for context-aware recommendation using differential relaxation [C]// ACM RecSys, the 4th Workshop on Context-Aware Recommender Systems. 2012.

[17] ZHENG Y, BURKE R, MOBASHER B. . Recommendation with differential context weighting [C]// The 21st Conference on User Modeling, Adaptation and Personalization. 2013: 152-164