目录

[问题概述 1](#_Toc9882052)

[1. 解决的问题 1](#_Toc9882053)

[2. 目前有哪些比较常用的方法 2](#_Toc9882054)

[3. 如何融合时间信息 2](#_Toc9882055)

[3.1. 基本定义 2](#_Toc9882056)

[3.2. BiasSVD算法 3](#_Toc9882057)

[3.3. 时间信息融入用户偏置项 4](#_Toc9882058)

[3.4. 时间信息融入物品偏置项 5](#_Toc9882059)

[4. 简单的结果验证 7](#_Toc9882060)

# 问题概述

推荐系统中的算法优化问题。协同过滤算法是实现推荐系统最重要的技术之一。随着时间的推移，用户对物品的偏好会不断的发生变化，物品自身的流行度也会随时间不断的发生变化。而目前常用的推荐算法如基于邻域的协同过滤算法itemCF、userCF和隐语义模型算法FunkSVD、BiasSVD、SVD++都没有将时间因素考虑进去。时间信息是一种非常重要的上下文信息，应该在算法中加以利用。本文使用Sigmoid函数和流行度函数将时间因素融入到了BiasSVD算法中，成功的设计出了一个融合时间信息的新算法Time-BiasSVD。在MovieLens数据集上的验证结果表明：该算法与已有协同过滤算法以及融合时间信息的算法timeSVD++相比，能更准确的预测用户实际评分，提高推荐系统的推荐质量。

# 解决的问题

随着大数据时代的到来，网络空间中蕴含的信息量几何式增长。在这种背景下，用户如何快速的获得自己需要的信息就变成了一个非常严峻的问题。推荐系统可以帮助用户发现他们可能喜欢的物品，这有效的解决了信息过载的问题。

协同过滤(collaborative filtering,简称 CF)是目前推荐系统中应用最广泛、最成功的技术[1]目前在工业界中得到最广泛应用的协同过滤算法是基于邻域的协同过滤算法。基于邻域的协同过滤算法包括两种，基于用户相似度[2]和基于物品相似度[3]。

隐语义模型(latent factor model,简称 LFM)使用用户的历史评分数据，挖掘出数据中隐含的特征[4]。找出用户与这些隐含特征的关系来构造用户-隐含特征矩阵。再找出隐含特征与物品之间的关系构造成隐含特征-物品矩阵。使用这两个矩阵来预测用户对物品的评分。

时间信息是一种非常重要的上下文信息，将时间信息融入推荐算法中是推荐系统要解决的重要问题之一。通常情况下，数据会跟随时间的变化而发生变化。因此模型能反映最新的数据表达的内容与特性并且能捕捉数据本质的长期趋势是推荐算法要解决的重要问题。针对这个问题产生了许多相关的著作[5,11]。

本文提出了一种在隐语义模型的基础上融合时间信息的新思路，并设计出了融合时间信息的协同过滤算法Time-BiasSVD。其核心思想是：通过构建一条权重变化曲线将用户兴趣按照评分产生的时间赋予不同的权重，使得每条记录对用户的影响不同。距离当前推荐时间越近的记录权重越大，对用户的影响越明显。而物品本身也存在不同时间段内流行度不同的问题，因此物品也是一个和时间相关的量。将Sigmoid函数与物品的流行度融入到当前流行的BiasSVD算法中就实现了本文的Time-BiasSVD算法。实验结果表明，该算法比已有隐语义模型算法能更有效地预测用户的评分。

# 目前有哪些比较常用的方法

Koren等人提出的timeSVD++算法[13]将物品偏差进行了分割，给每个分割段使用数值不同的物品偏差。将变成了。

与此同时timeSVD++算法也给出了一个使用一次函数给用户偏置项融入时间偏移量的方法，把用户偏置项也变成和时间相关的参数。将用户偏置项转变成了。公式为：

(1)

(2)

其中代表数据集中每条记录的评分时间，代表数据集中所有记录评分时间的平均值。和是两个参数，通过算法训练学习获得。

通过以上方式，timeSVD++较好的实现了在SVD++算法的基础上融合时间信息。

Liang 等人[14] 使用扩展函数维度和张量分解的方式实现了融合时间信息。

Cigdem BAKIR在SVD算法的基础上融合了时间动态信息[15]。在该算法中考虑了评分的“年龄”，即评分的时间到现在推荐时间的时间差。降低年老评分的权重，增加年轻评分的权重。关键公式如下：

(3)

在这个公式中代表评分的年龄，代表加了权重的评分的年龄，、是参数。

(4)

其中代表原始评分，代表权重评分。

孙光福等人[16]提出的SequentialMF算法使用用户的评分数据和评分时间信息来构建用户和产品的消费网络图，并根据图计算影响力最大的近邻集，再把近邻集用到概率矩阵分解模型中然后分别计算出用户和项目的特征向量，根据该特征向量预测重构评分矩阵，再对用户进行推荐。

与上述相关工作不同的是，本文在BiasSVD算法的基础上，创新的使用了变型的Sigmoid函数模拟用户兴趣偏置项随时间变化的趋势。又在项目偏置项中找出了电影流行度和平均评分的关系，进而将物品偏置项中的流行度分离出来，给物品偏置项也融入了时间信息。经过这两步最终就设计出了本文的融合时间信息的协同过滤算法Time-BiasSVD。

# 如何融合时间信息

## 基本定义

**定义1**（用户-物品评分矩阵）***R***={*r*u,i}，***R***表示用户对物品的评分矩阵。*r*u,i表示用户*u*对物品*i*的评分，为1-5之间的整数。

**定义2**（用户-隐含特征矩阵）***P***={*p*u,k}，***P***表示用户与隐含特征之间权重的矩阵。*p*u,k表示用户u对第k种特征的喜欢程度。取值范围为0-1之间的浮点数。

**定义3**（隐含特征-物品矩阵）***Q***={qi,k}，***Q***是表示物品在其所属特征中占重要程度的矩阵。qi,k表示物品i在特征k中所占的重要度。取值为0-1之间的浮点数。

**定义4**（用户偏置项）代表预测函数中用户的评分中用户自身性格等因素产生的误差。

**定义 5**（项目偏置项）代表物品本身属性导致的评分误差。

**定义6**（用户兴趣随时间跨度增大变化函数）代表一个物品对用户的刺激随着时间跨度的增大而呈现出的S型下降趋势。数据集中每个评分记录都会有一个时间戳，代表用户历史评分记录中的时间戳和当前为用户推荐物品的时间戳之间的差值。

**定义7**（物品在所属时间片内的流行度）代表物品在某一时间段内的流行度。代表某个具体的时间片。

**定义8**（物品的评分与流行度之间的关系）代表物品得到的评分与物品在某一时间段内的流行度之间存在的正相关关系。其中的就是定义7中的。

## BiasSVD算法

RMSE是根号下预测值与真实值之间差的平方与总预测次数N的比值。常被用来衡量协同过滤算法的好坏。FunkSVD的核心思想是实现RMSE在训练集上最小。

FunkSVD预测函数为：

(5)

代表预测评分。BiasSVD算法在FunkSVD算法的基础上添加了用户偏置参数和物品偏置参数。在下文中将使用来代表全局平均分，代表某个数据集上用户的平均评分标准。再将定义4和定义5中的参数加入之后，得到BiasSVD预测函数为：

(6)

为了得到向量p,q以及参数和，我们使用梯度下降算法最小化如下损失函数：

(7)

如果只用公式7来进行算法处理，又经常会遇到过拟合的问题。所以通常使用正则化方法来对参数bu、bi、qj、pi进行惩罚，损失函数公式最终如下：

(8)

BiasSVD中存在一个参数用来代表用户本身的偏置因素。而用户的性格和兴趣是的重要组成部分。通过前面的介绍可以知道用户的性格和兴趣会随时间的增长而发生变化。所以可以将参数融入时间信息，来作为实现Time-BiasSVD算法中的一部分。

## 时间信息融入用户偏置项

用户的评分习惯会随时间的推移而发生变化，以用户对电影的评分为例。

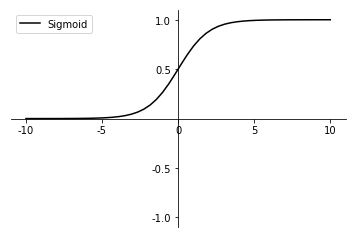
若干年前某用户观看了一部电影，这部电影给用户产生了一个刺激，这个刺激给用户造成的影响反映为用户对这部的电影的评分。但是随着时间的推移，用户看的电影变多导致用户阅历增加；同类型的电影看得太多导致对这类电影的新鲜度降低；经历的事情变多导致用户的兴趣发生变化。这些因素都会影响用户的评分习惯，最可能的表现是用户对电影进行打分的时候更加严苛。

在生态学中可以使用S型曲线来对孟德尔定律进行解释。即某种生物数量的增长随着时间的递增并不是呈J型增长，而是S型增长。因为当种群密度增长到某个数值之后，会遭受一系列客观因素的制约，导致其增长趋势受到制约，符合S型增长趋势。使用这个观点解释用户对电影的评分也是合理的。从现在的时间向前推，越接近评分记录产生的时间，这部电影给用户带来的刺激越大，但也不可能是呈现J型增长。因为在现在给用户推荐物品的时间点与评分记录产生的时间点之间，用户也会受到各种其他刺激的影响。本文使用变型的Sigmoid函数来描述这个变化趋势。

Sigmoid函数公式为：

 (9)

函数图像为：



**图 1** Sigmoid函数图像

Sigmoid函数有很多非常好的性质：它的定义域在(−∞,+∞)之间，值域在(0,1)之间，并且该函数在定义域内为光滑可导函数。

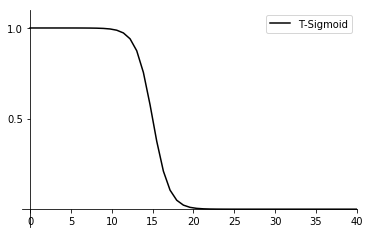
直接使用Sigmoid函数并不能满足我们的要求，因为本文需要的是电影给用户造成的刺激在第一象限内随时间跨度的增加呈现出S型递减趋势。因此本文把Sigmoid函数稍微做一下变型。Time-BiasSVD算法的变化函数公式如下：  
 (10)

将公式9对x轴做对称，之后向y轴正方向平移一个单位，再把整个函数图像向右平移个单位就可以得到公式10。

由于每个用户都是不同的，所以每个用户都有一个属于自己的和。可以决定S型函数从哪个点开始下降。决定S型函数下降部分的斜率，代表用户单位时间内兴趣变化的多少。

这两个参数都会在梯度下降算法中不断的迭代更新。代表用户评分记录中某条评分记录的时间戳，可以直接从数据集中获得。

经过变化之后，的一个实例图像如下：



**图 2** T-Sigmoid函数图像

经过以上的描述，可以构造出本文需要的，其公式如下：

(11)

使得的变化趋势与的变化趋势保持一致，且其定义域为[0,+∞ )，由于可能为0，所以其值域为[0,]。

通过使用公式11，可以将原BiasSVD算法的预测函数修改为：

(12)

## 时间信息融入物品偏置项

物品偏置项也是和时间信息有关的量。举例来说，周星驰的电影大话西游之大圣娶亲在刚上映的时候获得的评分并不高。但后来这部电影又重新被拿出来播放，大家给予的评价都非常高。因为这部电影描述的爱情故事正是现代人所崇尚的。 也就是说一部电影它讲述的故事和表达的观点在不同的时间段内被人们接受的程度是不一样的，这部电影的流行度在不同的时间段也是不一样的。本文认为某段时间电影的流行度会对电影的评分产生非常大的影响。

下面将对上述观点进行实例分析，以提前设置好的时间段参数为时间片对MovieLens数据集进行划分，可以根据实际情况进行选择和设定。选定了以后就需要计算每个时间片内每部电影的平均评分和流行度。

电影i的平均分计算公式如下：

(13)

其中代表第h个时间片内电影i获得的所有评分总和，代表第h个时间片内某部电影评分记录总数。

电影i的流行度计算公式如下：

= (14)

其中代表电影i在第h个时间片内被评价的次数，代表这段时间内所有电影被评价的总次数。

经过对数据集中所有电影的评分与流行度进行数据分析与绘图，可以得出结论：电影获得的评分与电影的流行度是正相关的关系。

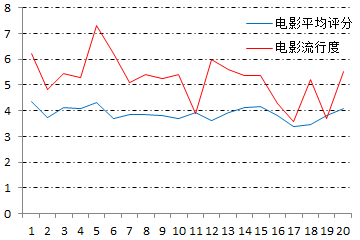
本文设置时间段参数为1周对数据集进行划分并以id为1的电影在不同时间片内的流行度和平均分制表，得到表1如下：

**Table1** Average score value change table

**表1** 平均分流行度数值变化表

|  |  |  |  |
| --- | --- | --- | --- |
| 时间片序号 | 电影平均评分 | 电影流行度 | 电影评价总数 |
| 1 | 4.375 | 0.006206 | 24 |
| 2 | 3.727 | 0.004811 | 11 |
| 3 | 4.133 | 0.005450 | 15 |
| 4  ┋ | 4.100  ┋ | 0.005285  ┋ | 10  ┋ |
| 28 | 3.808 | 0.003711 | 26 |
| 29 | 4.000 | 0.002922 | 5 |
| 30 | 4.100 | 0.005543 | 10 |

由于评价数量过少的记录不具备可靠性，因此将Table1中电影评价总数小于10的行删除。

使用平均分一列和流行度一列做折线图，为了便于观察将流行度一列扩大1000倍。折线图如下：

**图 3** 平均分与流行度变化函数图像

从图3可以看出，电影平均评分的曲线和电影流行度的曲线，变化趋势相似度非常高。通过图3，可以得到如下函数：

(15)

其中代表电影i在第h个时间片的流行度带来的评分。和代表是两个参数，在算法中迭代更新。表示电影在第个时间片的流行度。

但事实上，现实中也确实存在某部烂片由于大量炒作导致电影流行度很高，但实际评分很低的情况。本文给出的解决方案是设置合适的时间段参数，将适当的延长一些。这种炒作出来的烂片短暂的火爆过后，由于实际电影水平很差会导致流行度大大下降。所以只要选择一个合适的就可以将电影实际应有的流行度反映出来。

在公式6中有物品偏置项参数可以用来代表电影i所有方面的总偏置数值。在本文中将中与流行度相关的部分单独拆出来融入到用户预测评分函数中。

根据上文的描述将融合时间信息的物品偏置项参数设置为：

= + (16)

经过3.3节与3.4节的介绍，本文最终得出了Time-BiasSVD算法的预测函数为：

(17)

# 简单的结果验证

本文采用RMSE作为评价算法的标准。推荐算法整体的RMSE越小，意味着推荐算法的推荐质量越高。假设算法对N个产品预测的评分向量表示为{}，这N个产品的实际得分向量为}，RMSE的计算公式为：

RMSE = (18)

**实验结果与分析**

实验比较了五种算法在不同隐含特征参数下的结果。在实验中分别设定隐含特征数量为10、20、50。表2给出了不同算法在不同隐含特征参数下RMSE的结果，根据表2可以得出以下结论：

(1)随着隐含特征的增大，各个算法精度都有一定的提高。

(2)BiasSVD算法与FunkSVD算法相比，结果有了不小的提高，这说明在算法中融合偏置因素对结果的提升是有作用的。

(3)本文设计的Time-BiasSVD算法相比BiasSVD算法和FunkSVD算法以及SVD++算法在结果上都有比较大的提升，这说明本文提出的融合时间信息的方法可以有效的提高推荐精度。

(4)本文设计的Time-BiasSVD算法与Koren等人的timeSVD++算法相比，也提高了推荐精度。但是本文只与timeSVD++算法中最基本的模型进行了对比

。

**表2** 不同算法RMSE比较结果

|  |  |  |  |
| --- | --- | --- | --- |
|  | K=10 | K=20 | K=50 |
| SVD++  FunkSVD | 1.11379  1.09976 | 1.11293  1.09948 | 1.11041  1.09918 |
| BiasSVD  timeSVD++ | 1.09451 | 1.09266 | 1.09189 |
| Time-BiasSVD | **1.09182** | **1.09091** | **1.09047** |