**上海财经大学**

**本 科 生 毕 业 论 文**

**（请用黑色签字笔或钢笔填写）**

题 目：基于时效性和序列性的推荐系统分析

姓 名： 杨仕豪

学 号： 2015111476

专 业： 信息管理与信息系统

院、 系： 信息管理与工程学院

指导教师： 方慧

**基于时效性和序列性的推荐系统分析**

# **摘 要**

随着计算机技术和大数据技术的飞速发展，互联网上的信息大幅增加，造成了严重的信息超载。推荐系统便是解决信息超载的有效方案，在近三十年的飞速发展中，出现了基于用户或物品的协同过滤算法、基于语境感知算法等多种算法。本文介绍了三种基本的协同过滤算法，并利用亚马逊用户行为数据集和steam用户行为数据集进行了用户行为的分析。另外，传统的三种算法没有考虑时间数据，本文对临近模型进行了相似度计算的优化，并利用计算商品序列矩阵解决了商品购买序列性问题，两种方法的效果相较于临近模型都有所改进。最后本文实现了Koren文章中的奇异值分解方法，并将XGBoost算法和上文提到的考虑时间的临近模型算法并进行了综合分析比较。

关键字： 推荐系统 协同过滤算法 时效性 序列性 奇异值分解

Analysis of recommendation system based on

time effect and sequence

# **Abstruct**

With the development of computer technology and big data technology, data about users, commodities, transaction records, and social information on the internet has grown explosively, resulting in serious information overload. Recommendation system is an effective solution to information overload. In the rapid development of nearly 30 years, there are various algorithms based on user or item collaborative filtering algorithm, time-aware factor model and so on. This paper introduces three basic collaborative filtering algorithms, and the user behavior is analyzed using Amazon user behavior dataset and steam user behavior dataset. In addition, the traditional three algorithms do not consider time data, this paper optimizes the similarity matrix in the neighborhood model, and computes commodity sequence matrix to solve the problem of commodity sequence problem. The root mean square error of the two optimization methods on the Amazon dataset is reduced. Finally, this paper implements the SVD method in Koren article, and makes a comprehensive analysis and comparison between the XGBoost algorithm and the above time-considered model.

Keywords: Collaborative filtering commendation algorithm; Commodity sequence; Recommendation systems; SVD

目 录

[**摘 要** II](#_Toc8040444)

[**Abstruct** III](#_Toc8040445)

[第一章 绪论 1](#_Toc8040446)

[1.1研究背景和意义 1](#_Toc8040447)

[1.2推荐系统发展历史与研究现状 1](#_Toc8040448)

[1.3推荐系统的缺陷和问题 2](#_Toc8040449)

[第二章 推荐系统相关理论 3](#_Toc8040450)

[2.1基于协同过滤的方法 3](#_Toc8040451)

[2.1.1与基于内容的推荐方法的区别 3](#_Toc8040452)

[2.1.2临近模型 3](#_Toc8040453)

[2.1.3基线模型 4](#_Toc8040454)

[2.2隐式(SVD)模型和SVD++ 5](#_Toc8040455)

[2.3时间感知算法 6](#_Toc8040456)

[第三章 基于时效性和序列性的推荐算法研究 6](#_Toc8040457)

[3.1用户行为分析 6](#_Toc8040458)

[3.1.1数据集描述 6](#_Toc8040459)

[3.1.2用户描述 7](#_Toc8040460)

[3.1.3相关性分析 9](#_Toc8040461)

[3.2基于时间变化的协同过滤方法 10](#_Toc8040462)

[3.2.1时效性分析 10](#_Toc8040463)

[3.2.2推荐序列性 12](#_Toc8040464)

[第四章 实验与结果分析 15](#_Toc8040465)

[4.1实验准备及过程 15](#_Toc8040466)

[4.1.1研究方法 15](#_Toc8040467)

[4.1.2评价方法 15](#_Toc8040468)

[4.2算法比较 16](#_Toc8040469)

[4.2.1基于XGBoost的推荐算法 16](#_Toc8040470)

[4.2.2 实验结果 17](#_Toc8040471)

[第五章 总结与展望 18](#_Toc8040472)

[参考文献 20](#_Toc8040473)

# 第一章 绪论

## 1.1研究背景和意义

现如今信息在生活中扮演了越来越重要的角色，用户面对庞大的信息很有可能会束手无策或者花费更多时间来筛选信息，造成了愈来愈严重的“信息超载”。作为一个信息的生产者，如何了解消费者的偏爱和喜好，给予怎样的信息或产品才能更加满足消费者的需求，成为了生产过程中很重要的一环。作为一个信息的消费者，从众多信息中获取有意义的信息也是一件困难的事情。显然人们需要一个沟通用户和信息的桥梁，来帮助人们获得所需要的信息，推荐系统则搭建了这之间的一个桥梁。2003年IEEE Internet Computing上发表了一篇名为《亚马逊推荐系统二十年》的文章[[1]](#footnote-1)，回顾了亚马逊推荐系统二十年来的发展，亚马逊早已意识到推荐系统的重要性，上线了基于物品的协同过滤算法[[2]](#footnote-2)，并一直致力于构建一个千人千面的商店，即希望每一个打开亚马逊官网的人看到的内容和商品都不尽相同，因为呈现在每个人面前的商品都是亚马逊进行个性化筛选得到的结果。经历了近二十年的发展，推荐系统仍旧是互联网交易平台不可或缺的环节。

推荐系统是时代的产物，是信息爆炸下人们要求更高效率的产物。其产生背景一是存在信息过载，如果用户在庞大的信息中找到自己需要的信息不需要付出太大的代价，那么个性化推荐就失去了意义。第二则是用户并没有明确的可获取的需求，在这种情况下，只能靠预测来推断用户到底需要什么样的产品。

因为数据量的日益丰富，驱使当今的推荐系统不能只是简单的基于内容推荐，应用其中的应该是更加复杂的协同过滤模型、基于标签的模型、隐式语义模型等。在建立推荐系统的过程中，也遇到越来越多的问题，诸如消费时间是否能影响协同过滤中产品用户的相似程度、周期性的产品是否应该影响推荐系统的模式、推荐商品是否存在着因果关系等等问题。很显然我们发现，在推荐系统中，时间作为一个因素影响力也越来越大，对于亚马逊商品的数据集，本文会从基本的推荐算法入手，并对时间效应的几个不同方面对推荐系统进行优化和改进，并给予一定的可解释性。

## 1.2推荐系统发展历史与研究现状

推荐系统的概念自提出以来得到了广泛的应用，同时在商业领域也取得极大的成功。推荐系统根据成熟程度[[3]](#footnote-3)划分为三个阶段:

第一阶段，诞生阶段。Goldberg[[4]](#footnote-4)在1992年建立了最早的一个个性化邮件推荐系统Tapestry。它已经可以帮助人们过滤掉垃圾邮件，然后留下我们所感兴趣的邮件并依据重要性进行排序。其也引入了协同过滤的概念，利用其他用户对邮件的反馈来为用户筛选需要的符合条件的邮件。1994年明尼苏达大学双城分校计算机系的 GroupLens 研究组设计了名为 GroupLens 的新闻推荐系统[[5]](#footnote-5)，它是第一个自动化协同过滤系统。

第二阶段，推荐系统逐步实现了商业化。亚马逊在1998年首次提出了基于物品的协同过滤算法，充分利用了用户的历史行为数据进行分析，通过物品相似性来推荐用户可能感兴趣的商品。这种算法适用于物品繁多的场景，随即包括YouTube、Netflix在内的很多大公司都开始采用这种算法。

第三阶段，推荐系统在工业界有着十分优良的表现，人们利用用户历史行为数据开始对用户需求进行更深层次的挖掘和分析，并致力于在大数据背景下降低推荐系统的计算复杂度。机器学习和深度学习的高速发展也逐渐融入到推荐系统算法中来，文献[[6]](#footnote-6)利用了深度神经网路的系统过滤算法来做视频推荐，另外一篇文献[[7]](#footnote-7)利用卷积神经网络（CNN）和概率矩阵分解模型（PMF）结合提出了卷积矩阵分解模型。与此同时，大数据时代下时间信息也被人们记录下来并应用在推荐系统中，文献[[8]](#footnote-8)分析认为在音乐推荐过程中，某一天的特定时间人们可能更倾向于收听某种特别的音乐，比如说午后可能会喜欢民谣类型的歌曲。Koren在文献[[9]](#footnote-9)中提出了时间感知算法，把时间信息加入到了SVD推荐模型。

## 1.3推荐系统的缺陷和问题

时间在推荐系统中的作用随着数据维度的增加变得更加有意义，比如在运行协同过滤算法时，要计算两个物品相似度，传统算法并没考虑时间的因素，如果两个用户不同时间购买了同一个商品，传统的方法认为这对两个用户的相似程度影响相同，而事实上，如果购买的时间间隔越短，那么相似度理应越高。时间也是相似程度的决定性因素之一。另一个方面，商品购买序列对于用户而言也有一定的意义，比如，我们在购买手机之后可能会紧接着购买一款配套的手机壳，而不是在购买手机壳之后再购买手机。也就是说对于购买了手机壳的用户我们推荐手机是不合适的。另外，某些商品都有很明显的季节性或者周期性，像是电子产品则更新换代较快，推荐系统应该更加包容加入的新物品，对于周期性商品如何判断商品的周期并给予合适的推荐，也是推荐系统优化中很重要的一环。

# 第二章 推荐系统相关理论

推荐系统是利用用户的历史行为数据对用户未来可能购买的商品进行预测。历史数据可能包括了你的浏览记录、购买记录甚至是你朋友的购买记录。就点外卖这件事而言，第一种情况叫社会化推荐，就是我们可能直接询问朋友，哪一家外卖比较好吃配送较快等等，从而做出自己的判断。另一种方式叫做基于内容的推荐，比如我们在一家店订购过汉堡外卖，系统可能认为我的兴趣爱好就是汉堡，所以这次给我推荐了另外一家汉堡店的产品。当然还有一种更高级的方法，就是系统可以检索一下吃过汉堡的人都吃了一些什么，然后把这些东西再推荐给我，很大概率我和这些人有一定的共性，这种方式称为基于协同过滤的推荐。譬如还可以通过物品用户的隐含信息来进行推荐，这种方式称为隐语义模型推荐。本文主要对协同过滤算法进行探讨。

## 2.1基于协同过滤的方法

### 2.1.1与基于内容的推荐方法的区别

基于内容的推荐是推荐算法中最原始的算法。如果你浏览或购买过某种类型的商品，那么系统会推荐与其相同类型的商品。这种方式十分直观但是很明显新颖程度较差，一些比较昂贵的物品，比如说笔记本电脑，购买之后短时间内一般不会选择再次购买，所以这时基于内容的推荐就成了无用推荐。为了使推荐更具有多样性，基于协同过滤的推荐方法更多得利用了用户与用户之间、商品与商品之间的相似性并加权计算来做推荐。在对时间的处理上，基于内容的推荐也无法将时间信息考虑在内，导致了推荐系统不够完善。

在基于协同过滤的方法中，用矩阵U表示m个用户对n个物品的评分情况，分数越高表示越喜欢这个物品。表示用户u对物品i的打分情况。

### 2.1.2临近模型

ItemCF算法

Item算法就是基于物品的协同过滤算法，基本思想是在用户评分矩阵的基础上利用物品打分向量计算物品之间的相似性，然后把与用户喜欢的物品相类似的物品推荐给用户。Item算法第一步是计算各个物品之间的相似度，这也是体现基于物品的协同过滤算法优势的一点，物品的性质不会因为时间的变化而变化太大，所以物品的相似度比较稳定。计算相似度常用余弦相似度计算：

其中表示物品i和物品j的打分向量。

或者可以使用皮尔逊相关系数来计算相似度：

其中表示用户u对物品i的打分，表示第i个物品打分的平均值。

计算得到了物品之间的相似度，相似度代表了每个评分的重要程度，相似度越高则越重要，即用对用户u已打分的物品的分数进行加权求和，权重为这些物品与要评分物品之间的相似度，得到预测公式如下：

其中表示用户u对物品i的预测打分，K表示与i相似度高的物品的集合，这个集合很容易从相似度矩阵中得到。

UserCF算法

UserCF算法和Item算法很大程度上是一样的， 但他们的思想是恰恰相反的。具体算法和Item算法没有太大差别，所以这里主要阐释一下其区别：UserCF算法因为是基于用户的，所以在一群兴趣爱好相似的人群中可能会有更好的表现。ItemCF着重于维护每个用户本身的特点和喜好，它是在每个用户自己已经打分物品的基础上进行推荐的，推荐结果更加贴近用户喜好。UserCF要必须计算用户的相似度矩阵，所以在用户数量较少时会有更低的复杂度，ItemCF则要计算物品的相似度矩阵，在物品种类较少时会有更低的复杂度。

### 2.1.3基线模型

在一个庞大的数据集中，会有百万级的用户和商品，在打分程度方面每个人也有每个人的风格，比如一个人性格比较挑剔所以给予商品的评分会普遍较低，而另一个人可能给分普遍较高，但这不能完全代表这两者不相似，所以模型需要尽量消除这种评分偏差带来的影响。假设代表总体打分均值，用表示这种基于偏差的估计：

其中，和表示用户u和物品i打分时的偏差。用来估计基本避免了因为打分习惯引起的相似度误差，可以用最小二乘法对和进行最优化：

其中，第一项是寻找最优的和，第二项是对和的惩罚项以防止过拟合。

## 2.2隐式(SVD)模型和SVD++

在Koren的文章中，他们在解决Netfilx问题时用到的SVD++模型，就来自于隐式模型。隐式模型利用了用户和物品的隐变量属性进行建模，隐变量在一定程度上反映了物品的属性特征和人物的喜好特征。例如对于电影产品而言，隐变量代表了电影的类型、导演级别、出版社级别等等。用和和来表示物品和人物所对应的隐变量，分别对应了电影和用户在各个隐藏特质上的特征向量。在不考虑偏差时可以直接用和对进行估计：

如果将用户和电影本身评分偏差的影响考虑进来，那么我们可以将SVD模型与基线模型结合得到：

进而转化为了优化问题：

对于每一个给定的打分，用来表示误差，可以用梯度下降方法或迭代的最小二乘算法求解。在迭代最小二乘算法中，首先固定优化，然后固定优化，交替更新，并用下面的迭代方式更新参数直到收敛:

γ是更新步长，在求解过程中，利用参数γ=0.005,λ=0.02。

上述模型被称为SVD模型，基线模型包含了用户和商品的偏好信息，临近模型包含了用户、商品之间的相关性信息，而SVD模型包含了用户和商品的潜在属性。与此同时，Koren在Netflix比赛中使用的方法是在这个模型中又加入了隐式反馈来对进行补充, 可以把看作是显示反馈，它有很好的可解释性，比如代表用户某个方面的喜好，整个SVD++模型可以看作是显示反馈、隐式反馈和偏见偏好的结合，其中隐式兴趣对应的向量也是 k 维，它由用户有过评分的物品生成:

其中表示用户u的打分集合，隐式反馈用表示，一般是通过用户购物历史获得，所以遍历.

## 2.3时间感知算法

在算法介绍中，SVD算法将用户的喜好和商品的特征进行分解，用来表示预测的结果，当加入基线模型时得到：

进而转化为了优化问题：

而时间感知算法认为，人们的兴趣喜好是随着时间变化的，比如高中时期人们可能比较喜欢书籍教辅资料，等到上了大学人们的喜好就不再是这些书籍，反而是电影游戏这一类。另一个方面，物品的特征是不会显著变化的。这种算法就是把每一个变量看作是随时间变化的变量，Koren在提出了SVD模型之后，加入了时间信息：

其中表示削减常量，表示用户u平均的打分时间，和可以通过线性回归得到估值，中表示将时间t进行等分，时间段的个数就是取值的大小。

# 第三章 基于时效性和序列性的推荐算法研究

## 3.1用户行为分析

### 3.1.1数据集描述

本文采用了两个主要数据集:亚马逊公开用户商品打分数据集和steam用户行为数据集。亚马逊提供了总计142.8 m的用户打分评论数据，并依据商品种类对数据集进行划分，本文利用了总计2,252,771条数据来做用户行为分析和推荐系统设计。数据由用户编号，商品编号，用户评分和时间戳组成。因为包含了时间戳，所以更容易进行时效性方面的分析。

Steam用户行为数据集的选取是因为相比庞大的亚马逊数据集，steam游戏商品没有严格的细分种类，一个用户可能拥有很多品类的游戏，一个游戏也会被多个玩家购买，所以在推荐的过程中更具有代表性，这个数据集包含了20万条用户购买使用数据，包括了用户ID，游戏名，是否购买和游戏时间四条数据。

### 3.1.2用户描述

首先分析原始数据，看总体用户的打分习惯，利用python对1到5分打分的用户数量进行汇总得到：

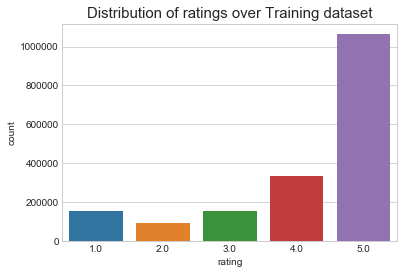


图 1 用户汇总

通过图1可以看出过半数的用户商品评分达到了五分，可见亚马逊玩具商品总体质量还是很不错的，其次1分商品的数量大致占总体的9%且略高于2分商品，侧面说明亚马逊用户评分带有极端性，即可能在对商品比较满意时给予最高分数同时在对商品不满意时给予最低分数。因为原始数据还给了评分时间的信息，利用python将时间戳格式转化为时间格式，并对星期进行汇总（图2）:

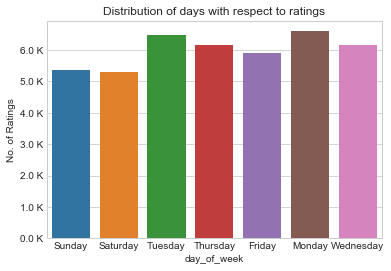


图 2 按周分类的用户行为数据

周六和周日用户评分数量明显少于工作日，而在工作日中周一和周二的用户评分量较高。其中蕴含了周期性规律，当然它不一定是以周为周期，还可能包含了其他的规律，时间周期也是影响推荐系统效率的重要因素。

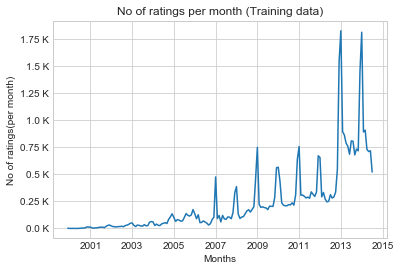


图 3 评分数量随时间的变化

图3为按照月份将评分数量汇总得到的，很明显看出2006年之前的评分数量整体都比较少，也没有表现出很强的季节性。2007年春天的评分数量开始激增并随之恢复正常值，在接下来的几年当中，在平均评分数量缓慢增长的情况下呈现出一定的周期性，主要体现在每年12月到2月份会有明显的销量上升，内在原因可能有二，一是每年冬末春初之际，该类型商品都会有较大的折扣导致销量上升评分数量增多，二是人们对于商品的需求本就有季节特征或周期性。

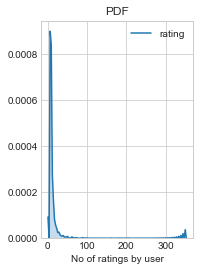


图 4 商品购买数量概率分布图

通过绘制所有用户购买商品数量PDF图4，可以看出大多数用户购买商品的数量围绕在1.3个附近，购买超过5个的用户数量骤降，但有一少部分用户会购买非常多的商品。

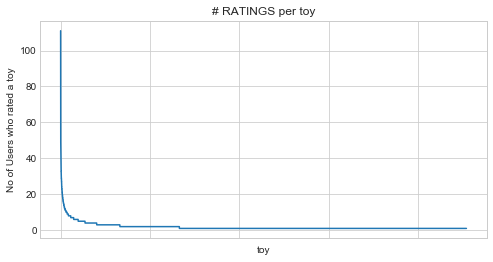


图 5 商品购买频数分布

从商品角度来看，商品购买频数如图5所示，极少部分的商品是热门商品。在推荐系统设计中，应该比较重视这部分商品的处理，因为其本质可能更加大众化，这使得我们推荐这些热门商品的优先级降低了，所以可以用修改权重来实现。

### 3.1.3相关性分析

相关性分析利用的是steam用户行为数据集，首先对得到的数据进行去重处理，并得到用户商品打分矩阵（这里是用户商品使用情况矩阵，如游戏购买则为1，未购买则为0）。这里使用itemCF的思想，利用余弦相关系数计算出每个游戏之间的相关性，得到相关性矩阵。

进一步的，可以对每一款游戏相关的游戏进行相关度排序，进而得到一个相关游戏的集合，以一款名为“Borderlands DLC The Secret Armory of General Knoxx”的游戏为例，与这款游戏相关度最高的游戏集合为（Table3）:

|  |
| --- |
| Borderlands DLC The Zombie Island of Dr. Ned |
| Borderlands DLC Mad Moxxi's Underdome Riot |
| Borderlands DLC Claptraps New Robot Revolution |
| Borderlands |
| Borderlands 2 |
| Red Faction Armageddon |
| Metro 2033 |
| Batman Arkham Asylum GOTY Edition |
| Deus Ex Human Revolution |

Table 实验结果

很容易看出与这个游戏最相关的五款游戏中，无一例外都是“Borderlands”系列的游戏。也就是说，用户在购买这款游戏时一般会打包购买这几款十分相关的游戏，其余弦相似度达到了0.98。从一个角度考虑这可能为推荐系统提供了极大的便利，因为相似度高正是推荐商品所需要的。但另一个角度，这些相似度极高的“系列产品”在一定程度上并不需要我们推荐，因为用户大概率会选择直接打包购买，相反相关度较高的“Red Faction Armageddon”虽然相似度只有0.48但是更值得推荐。

## 3.2基于时间变化的协同过滤方法

从上述分析中可以看出，商品的流行具有一定的季节性和周期性，同时相关程度和是否值得推荐之间存在着一定的关系。本文从时效性和序列性方面对问题进行了进一步的探讨。

### 3.2.1时效性分析

时间在推荐中占有很重要的地位，同时人的兴趣是变化的，用户不可能永远只购买某一类或者某一个商品，同时物品也有可能会在一段时期内很“火爆”而另外一些时间面临着冷清甚至是淘汰，所以整个推荐系统变成一个动态的结构。亚马逊用户行为数据集是一个包含了用户u在时刻t对物品i产生评分为d的行为信息。

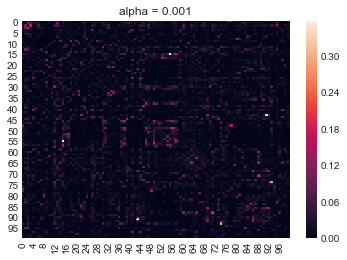
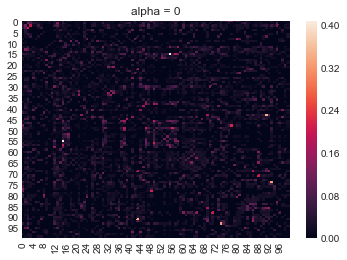
因为有了时间信息，所以用户每次登陆时，系统都会给予用户一个实时的推荐。这个推荐既是给予了他的购买历史给予的推荐又考虑了当前时间应该推荐那些商品。与此同时时间因素的部分又不能占有太高的比例，因为从短期来看，维持用户的喜好是很重要的，用户的兴趣爱好是推荐的基础。为了保证推荐系统的时间多样性，在推荐系统中尽量实现两点，第一是维持用户的兴趣爱好，每次推荐都能根据用户最近的行为给出一定的推荐，同时如果用户最近并没有购买行为或者浏览行为，推荐的结果也是会轻微变化的，而这些变化主要来自于时间。

在基于物品的协同过滤算法中，它通过余弦相似度计算物品的相似度：而在给用户u做推荐时，通过计算用户u对物品i的预期打分。

在有时间信息的情况下，相似度会随之发生改变，定义一个衰减函数来表示这种衰减，得到修正后的公式：

在公式中加入了时间衰减项，表示用户对物品i和物品j产生购买行为时间跨度上距离越远，那么函数值越小，进而相关性也就越小。是用户u对物品i产生购买或浏览行为的时间节点。是时间衰减系数，它的取值影响了整个系统对时间因素的敏感程度。如果用户的兴趣变化较明显，的值就应当较大，反之较小 。表达式中，表示当前时间,越靠近 ，物品i和物品j相似度就会有较高的权重，在结果中也会表现越突出。

利用亚马逊游戏用户行为数据（包含1048576条数据）得知，游戏的种类比玩具少，这样游戏之间的相关性更容易体现出来。当我们只用是否购买作为衡量评分的标准时的商品相关性矩阵：



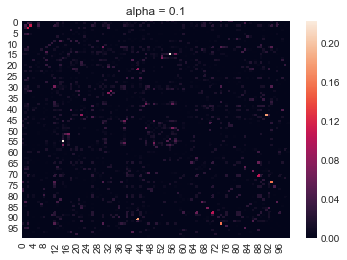
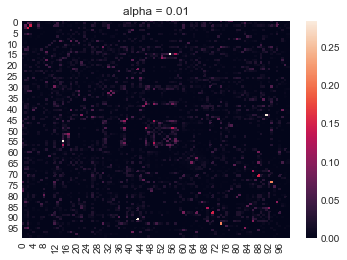
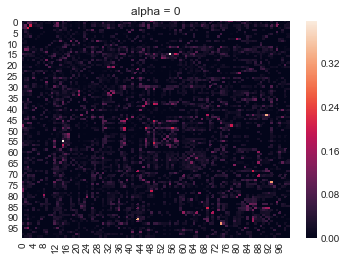
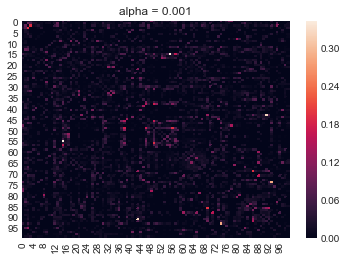


图 6 相关性矩阵热力图

图6中，我们发现当较大时，此时时间效应对相关性的削减很强，导致整体相关度都较弱，时，得到的也就是传统的余弦相关性矩阵，此时相关度最高达到0.408。当使用评分作为标准衡量商品相关性时，得到图7：



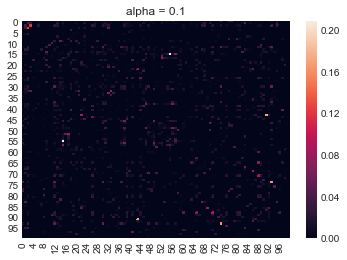
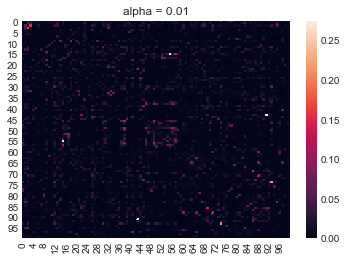


图 7 相关性矩阵热力图（以评分衡量）

这里使用了评分作为相关度计算标准，所以相关性整体会下降。因为相关性和最终预测函数中加入了时间约束，所以在预测过程中会出现结果随着时间变化而变化的现象，以某一商品为例。

公式中的遍历2001年到2019年这十八年得到的预测结果如下图8。红色线表示，蓝色线表示.

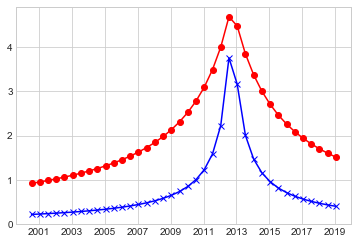


图 8 评分预测随时间变化曲线

若从此刻来看，最适宜推荐给用户该商品的时间是2012年6月份左右，随着时间的延续，预测的分数逐渐降低并且下降梯度减少，也就是说评分预测的变化受到时间距离最佳推荐时间的长度增长而越来越少。同时纵向来看，对于推荐的峰值大致削减了26%，而随着时间远离这个最佳推荐时间，削减程度更加明显。

### 3.2.2推荐序列性

日常生活购物过程中，我们会发现有时物品的购买具有序列性，例如我们在购买手机之后可能会紧接着购买一款配套的手机壳，而不是在购买手机壳之后再购买手机。也就是说对于购买了手机壳的用户我们推荐手机是不合适的。包括书籍、电影和电视剧都有这样的特征，在之前steam用户行为数据中，给用户推荐合适顺序的游戏序章是极为重要的，这不仅是有助于增加销售量，用户的体验也十分好。

为了探寻这种序列性，数据中的时间数据是很关键的，定义一个新的布尔变量表示用户是否购买过物品对并且是有序的，即用户u先购买了物品i然后购买了物品j，定义表示用户u购买物品i 和物品j的时间间隔。

以亚马逊video game用户行为数据为例，构建一个商品序列矩阵M

对于构建好的矩阵进行数据统计：

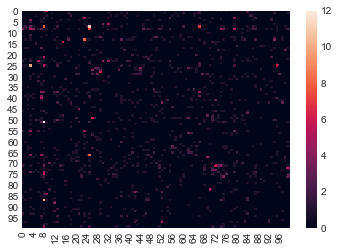
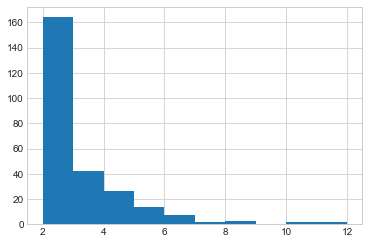


图 9 序列对出现次数频数柱状图和热力图

图9左图中横轴表示序列对出现的次数，即，纵轴表示次数的频数。右图表示序列对出现次数的热力图。很显然在样本中购买同一序列对商品的用户数量的频数随着用户数量的增加而减少。并且在右图中，并没有表现出很明显的对称性，也就是说可能某些序列对存在更深的意义。

另一方面，序列对和序列对的用户购买数量差值也能反映出购买商品时序列的意义，比如说，以及，那么说明样本中大部分用户在购买i之后会购买j，而小部分人会再购买j之后再购买i，这就体现了推荐序列性的意义。

已经建立好了商品序列矩阵，矩阵中数值的大小代表着序列对的强弱（强意味着用户更愿意按照这个顺序购买商品i和商品j）进一步的，对于按照序列对购买的用户u，可以计算出购买的时间差，一般来说，购买i和j时间间隔越长，他们的相关性就越低，反之越高。利用指数函数表示这种削减,构建一个新的矩阵：

表示用户购买的时间间隔，利用作为削减系数，其大致图像为图10：

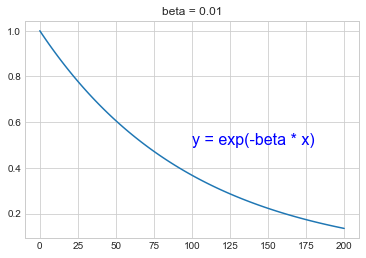


图 10 函数示意图

这里采用0.0146，也是体现序列关系的矩阵，数值的大小表示在指数权重下的时间期望，时间越长意味着两个物品相关性越低，时间越短反之。因为得到的T仅仅记录了商品之间的关系，这种关系是固有的，比如说相机和储存卡之间本身就存在着先后购买的因果关系，而不是人为给定的，所以在临近模型中可以加入这个时间系数作为最终预测的修正：

表示用户u购买的商品集合，是修正系数，表示序列补偿函数，在预测用户u对商品i的打分时，不仅利用了与i相关的商品集合中商品的相似度，并且考虑所有用户u购买的商品中是否有和商品i有很强烈序列关系的商品，如果i和j构成的序列对在有较高的表现，那么意味着大部分人可能在购买j之后会购买i，那么i应该更需要被推荐给用户u.

当然也可以用百分比来表示这种序列关系，通过计算购买某一商品的用户中有百分之多少的用户在后来购买了另一个商品，比如在样本中在用户购买77号商品之后在购买40号商品的条件概率是22.58%。但是这个概率仅仅包含了时间的先后，并没有涵盖时间的长短，举个例子是两个不同时期的热门商品，如果所有的用户在这两个时期都选择购买，那么计算出的百分比是百分之百，这样显然就失去了意义，而通过计算期望购买时间差的方法就可以有效避免这个问题。

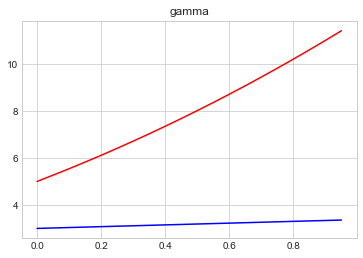


图 11 Gamma值变化对评分预测的影响

在算法设计中，相似度依旧是用了时效性分析中的相似度矩阵，但是在预测过程中加入了时间序列的考虑，上图11中横轴表示gamma的变化，纵轴展示了两个商品的预测评分的变化，因为这里预测值可能乘以多个大于1的值，所以最终的结果很可能会大于5（评分上界），但是其实对于商品的推荐并没有太大的影响，依旧会选出分值较高的纳入推荐范围。Gamma值反映了商品的序列性在最终结果中的影响大小，红色线对应的商品显然存在于很多序列对中使得评分会随着gamma的增加而显著提升，蓝色线对应的商品是更加独立的，增加序列性的影响并不能使最终结果有显著变化。

# 第四章 实验与结果分析

## 4.1实验准备及过程

### 4.1.1研究方法

首先收集合适的数据集，本文主要利用了亚马逊商品用户（video game dataset）评分数据集进行实验，这个数据集有1048576条行为数据，其中包含了652262个用户和41360个游戏。因为数据量较大并且十分稀疏，所以要进行一定的筛选，本文首先从原始数据中选取了较为活跃的用户和游戏，选择参与评价的游戏在5个以上的用户，和评分数量高于50游戏。然后选择2009年12月1日之后的所有数据作为新的数据集。

新的数据集中包含227142条数据，训练及和测试集数据比为4：1，分别是181713条和45429条。

|  |  |  |
| --- | --- | --- |
|  | 训练集 | 测试集 |
| 评分数 | 181713 | 45429 |
| 用户数 | 45607 | 14579 |
| 游戏数 | 3000 | 1762 |

Table 训练集和测试集数据描述

本文首先通过python实现基本的协同过滤算法，进一步使用机器学习XGBoost算法进行分析实验，最终将时效性和序列性分别与基础算法结合，得出相应的结论。

### 4.1.2评价方法

本文主要使用离线测试的方式对推荐系统算法的效果进行评测，在考量实验结果和真实结果的差别时，一般用平均绝对误差(Mean Absolute Irrorate)或者均方根误差(Root Mean Square Error RMSE)来衡量推荐的准确率。

MAE, RMSE定义如下:

T表示用户商品集合。这是对于评分结果的评测。通过计算精确率、召回率、F1来衡量推荐算法的效果，一般情况下精确率和召回率有相反的表现状况，Fl值则会同时考虑这两个指标，定义如下:

表示推荐给用户的商品列表，表示用户实际购买的或者实际感兴趣的商品列表。

## 4.2算法比较

### 4.2.1基于XGBoost的推荐算法

在临近模型中，我们利用用户对与被打分物品相关的物品打分情况来获得评分情况，而这些评分是有助于利用机器学习的模型预测打分，而不是直接通过公式来预测。

首先，建立用户相似度矩阵和商品相似度矩阵，如果要预测用户u对商品i的评分，找到和i最相似并且u也购买过的商品的评分并作为学习模型的特征，再找到和u最相似并且购买过的商品i的评分并也作为学习模型的特征。构建好的新的数据集如下表4：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| user | game | GAvg | sur1 | sur2 | sur3 | sur4 | sur5 | smr1 | smr2 | smr3 | smr4 | smr5 | UAvg | MAvg | rating |
| 0 | 0 | 4.0 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 4.6 | 4.48 | 5 |
| 27 | 0 | 4.0 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 4.5 | 4.48 | 5 |
| 338 | 0 | 4.0 | 5 | 5 | 5 | 5 | 5 | 5 | 3 | 4.3 | 4.3 | 4.3 | 4.3 | 4.61 | 5 |
| 409 | 0 | 4.0 | 5 | 5 | 5 | 5 | 5 | 1 | 2 | 1 | 1.2 | 1.2 | 1.2 | 4.48 | 1 |
| 621 | 0 | 4.0 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4.0 | 4.48 | 5 |

Table XGBoost训练集

其中GAvg表示所有打分的平均值，sur1-sur5表示与用户u最相似（取前五）的用户对i的打分，smr1-smr5表示用户对与商品i最相思（取前五）的商品的打分，UAvg表示用户u所有打分的平均值，MAvg表示商品i所有打分的平均值。一共有十三个特征值。

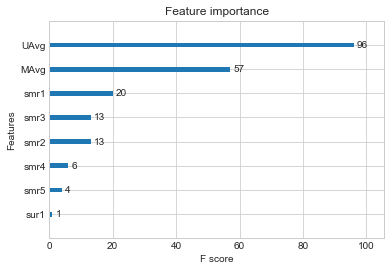


图 12 重要性分析结果图

如图12利用xgb内建变量重要性来看这十三个变量，UAvg用户个人打分均值重要性最高，MAvg商品打分均值其次。作为一个机器学习模型，可以允许加入更多的特征进行训练，这也就可以让XGBoost和其他协同过滤的训练方法结合，得到综合模型。

### 4.2.2 实验结果

本文中的实验在Intel®Core(TM)i7 2.40GHz 处理器，内存8G，windows操作系统，python3.5编译下完成的。实验结果采用了RMSE指标：

|  |  |  |
| --- | --- | --- |
| 算法名称 | 模型名称 | RMSE |
| XGBoost | first\_algo | 0.650002 |
| 基线模型 | bsl\_algo | 1.13832 |
| SVD++模型 | svdpp | 1.15441 |
| SVD模型 | svd | 1.16129 |
| 基于用户的临近模型 | knn\_bsl\_u | 1.22062 |
| 基于物品的临近模型 | knn\_bsl\_g | 1.22062 |

Table 基础模型实验结果

表5展示的是在steam\_video\_game数据集上的实验结果，其中XGBoost的RMSE最小，其次是基线模型。因为XGBoost的效果较好，所以把协同过滤算法的结果加入到XGBoost的特征中来得到表6：

|  |  |  |
| --- | --- | --- |
| 算法名称 | 模型名称 | RMSE |
| XGBoost+基线模型+基于物品的临近模型+SVD模型 | xgb\_final | 0.647133 |
| 基线模型+ XGBoost+基于物品的临近模型 | xgb\_knn\_bsl | 0.647133 |
| 基线模型+ XGBoost | xgb\_bsl | 0.648695 |
| XGBoost | first\_algo | 0.650002 |
| 基线模型+基于物品的临近模型+SVD模型 | xgb\_all\_models | 1.11609 |
| 基线模型 | bsl\_algo | 1.13832 |
| SVD++模型 | svdpp | 1.15441 |
| SVD模型 | svd | 1.16129 |
| 基于用户的临近模型 | knn\_bsl\_u | 1.22062 |
| 基于物品的临近模型 | knn\_bsl\_g | 1.22062 |

Table 综合模型实验结果

从实验结果可以看出，XGBoost实验效果不论是在训练集还是测试集都有比较好的效果，在加入了新的特征之后RMSE也有进一步的提升，用XGBoost自带的重要性评判中（如图7）：

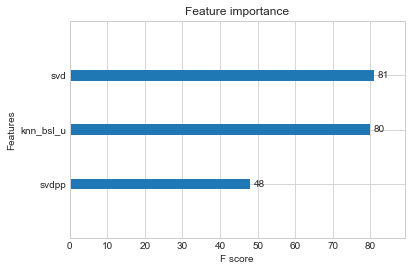


Table 重要性分析结果图

在临近模型和SVD模型相比，SVD具有更高的重要程度，其次是基于用户的临近模型。

在第四章中考虑了时效性和序列性在推荐系统中的影响，思想是在临近模型的基础上对相关性进行修正和对序列性进行研究，实验结果如表8：

|  |  |
| --- | --- |
| 模型名称 | RMSE |
| 基于物品的临近模型+时效性分析 | 1.198066 |
| 基于物品的临近模型+序列性分析 | 1.213142 |
| 基于物品的临近模型+时效性分析+序列性分析 | 1.189296 |
| XGBoost+时效性分析 | 0.647029 |
| XGBoost+序列性分析 | 0.647462 |

Table 加入时效性考虑的综合实验结果

一方面序列性分析在临近模型中单独表现并不突出，但是他有很强的可解释性，因为本文采用的数据集具有一定的局限性，在商品序列表现上没有达到很好的效果，另一方面，时效性分析对于临近模型有很好的改善，二者的结合也有比较好的效果。总体来看，时间的因素的考虑是必不可少的，在未来会有更大的时间跨度和更精细的时间节点，这会使时效性分析和序列性分析有更好的发展空间。

# 第五章 总结与展望

本文首先推荐系统历史发展状况和实现意义，进一步介绍了主流的三种推荐方式，考虑到时间因素，对临近模型在时效性、序列性方面进行了探究，并对用户商品打分在时间维度进行了分析和总结。最后利用python实现了Koren论文中SVD++算法并进行了多角度的算法准确率比较。

就时效性而言，除了时间对相关性的影响，商品序列性之外，还有很多方面有待完善。我们发现一本书的购买信息可以暴露很多用户的兴趣，让我们能够给出很多高度相关的推荐。这时候基于物品的协同过滤可能会格外有效。换一个场景，比如购买一款游戏时，可能会有很明显的购买先后顺序，这时候序列性又能有比较好的表现，但是这一算法不适用书籍类产品。另一个方面，时间相关性和序列性其实是矛盾的，时间相关性希望用户可以购买更加广泛的商品，而序列性则要求用户专注于某一领域的购买，当用户很明显是在寻找某个具体的商品时，推荐系统应该减小范围帮助用户快速找到所需要的商品。但是当意图并不确定时，探索性和新奇性应该是推荐的目标，找到推荐系统中多样性的正确平衡点，这也是至关重要的。

未来的推荐系统一定会朝着这四个方向发展，简单、可扩展，经常能给出令人惊喜和有用的推荐，可根据用户的新信息立刻更新推荐，可解释性强。未来的推荐会在智能计算机发展基础上更加的造福于人类。

# 参考文献

[1]G. Linden, B. Smith, and J. York, “Amazon.com Recommendations: Item-to-Item Collaborative Filtering,” IEEE Internet Computing, vol. 7, no. 1, 2003.

[2]G.D. Linden, J.A. Jacobi, and E.A. Benson, Collaborative Recommendations Using Item-to-Item Similarity Mappings, US Patent 6,266,649, to Amazon.com, Patent and Trademark Office, 2001 (filed 1998).

[3]朱扬勇，孙蜻.推荐系统研究进展[[J].计算机科学与探索,2015,9(5):513-525

[4]Goldberg D, Nicholls D, Brian M.O, et a1.Using collaborative filtering to weave an information tapestry[J].Communications of the ACM一Special issue on information filtering,1992,35(12),61-70.

[5]Resnick P, lacovou N, Suchak M, et al. Group Lens : an open architecture for collaborative filtering of netnews[C].CSCW 1994 proceedings of the 1994 ACM conference on Computer supported cooperative work, 1994: 175-186

[6]Xin D, Lei Y, et a1.A Hybrid Collaborative Filtering Model with Deep Structure for Recommender Systems[C]//Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence,2017:1309-1315.

[7] Kim D , Park C , Oh J , et al. Convolutional Matrix Factorization for Document Context-Aware Recommendation[C]// Acm Conference on Recommender Systems. ACM, 2016.

[8] Dias R , Fonseca M J . Improving Music Recommendation in Session-Based Collaborative Filtering by Using Temporal Context[C]// IEEE International Conference on Tools with Artificial Intelligence. IEEE Computer Society, 2013.

[9] Koren, Yehuda , and R. Bell . "Advances in Collaborative Filtering." Recommender Systems Handbook (2015):145-186.

1. [1]G. Linden, B. Smith, and J. York, “Amazon.com Recommendations: Item-to-Item Collaborative Filtering,” IEEE Internet Computing, vol. 7, no. 1, 2003. [↑](#footnote-ref-1)
2. [2]G.D. Linden, J.A. Jacobi, and E.A. Benson, Collaborative Recommendations Using Item-to-Item Similarity Mappings, US Patent 6,266,649, to Amazon.com, Patent and Trademark Office, 2001 (filed 1998). [↑](#footnote-ref-2)
3. [3]朱扬勇，孙蜻.推荐系统研究进展[[J].计算机科学与探索,2015,9(5):513-525 [↑](#footnote-ref-3)
4. [4]Goldberg D, Nichols D, Brian M.O, et a1.Using collaborative filtering to weave an information tapestry[J].Communications of the ACM Special issue on information filtering,1992,35(12),61-70. [↑](#footnote-ref-4)
5. [5]Resnick P, lacovou N, Suchak M, et al. GroupLens : an open architecture for collaborative filtering of netnews[C].CSCW 1994 proceedings of the 1994 ACM conference on Computer supported cooperative work, 1994: 175-186 [↑](#footnote-ref-5)
6. [6] Xin D, Lei Y, et a1.A Hybrid Collaborative Filtering Model with Deep Structure for Recommender Systems[C]//Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence,2017:1309-1315. [↑](#footnote-ref-6)
7. [7] Kim D , Park C , Oh J , et al. Convolutional Matrix Factorization for Document Context-Aware Recommendation[C]// Acm Conference on Recommender Systems. ACM, 2016. [↑](#footnote-ref-7)
8. [8] Dias R , Fonseca M J . Improving Music Recommendation in Session-Based Collaborative Filtering by Using Temporal Context[C]// IEEE International Conference on Tools with Artificial Intelligence. IEEE Computer Society, 2013. [↑](#footnote-ref-8)
9. [9] Koren, Yehuda , and R. Bell . "Advances in Collaborative Filtering." Recommender Systems Handbook (2015):145-186. [↑](#footnote-ref-9)