

# 南京信息工程大学

## 本科生毕业设计



题    目 基于用户和物品的电影推荐系统

学生姓名 \_\_\_\_\_

学    号 \_\_\_\_\_

院    系 人工智能

专    业 \_\_\_\_\_

指导教师 \_\_\_\_\_

二〇二〇年四月二十四日

# 声 明

本人郑重声明：

- 1、 以“求实、创新”的科学精神从事科学研究工作。
- 2、 本论文中除引文外，所有测试、数据和相关材料均为真实有效的。
- 3、 本论文是我个人在指导教师的指导下进行的研究工作和取得的研究成果，请勿用于非法用途。
- 4、 本论文中除引文和致谢的内容外，并未抄袭其他人或其他机构已经发表或撰写过的研究成果。
- 5、 关于其他同志对本研究所做的贡献均已在论文中作了声明并表示了谢意。

作者签名：

日期：二〇二〇年四月二十四日

# 目 录

1 绪论 .....	1
1.1 研究背景及意义 .....	1
1.2 国内外研究现状 .....	3
1.3 论文主要工作 .....	5
1.4 论文主题结构 .....	5
2 关于推荐算法的研究 .....	6
2.1 有关推荐算法的相关简介 .....	6
2.2 实现系统的算法介绍 .....	7
2.3 相似度计算 .....	9
2.4 如何测评推荐算法 .....	10
3 推荐系统的实现技术研究 .....	11
3.1 实验环境介绍 .....	11
3.2 系统实现相关技术介绍 .....	11
3.3 系统的设计 .....	11
3.4 电影的数据采集系统的实现 .....	12
3.5 URL 构造策略 .....	13
3.6 数据提取与存储采集 .....	13
4 推荐系统的设计与实现结果 .....	14
4.1 数据爬虫模块的实现 .....	14
4.2 需求分析 .....	15
4.3 用户需求模块 .....	15
4.4 系统实现 .....	15
4.5 系统功能模块简述 .....	15
4.6 数据库逻辑结构展示 .....	21
4.7 系统 E-R 图展示 .....	22
5 总结与展望 .....	22
5.1 总结 .....	22
参考文献 .....	23

致谢 .....	25
----------	----

# 基于用户和物品的电影混合推荐系统

**摘要：**本文设计了一个非常个性化的电影推荐系统。正如大众所熟知的那样，设计出一个高效的个性化电影推荐系统帮助人们解决电影搜寻难题是十分有必要的。因此本文的目的就是为了使用户找到自己感兴趣的内容进而设计的推荐系统。

这个系统计划采用网络爬取的数据集，首先将数据集包含的全部文件存储到数据库中并划分成测试集与训练集，进行分析生成电影推荐列表。

本文还介绍了一些基本推荐算法原理和基本公式，本系统还分析阐释了推荐系统的结构及各功效。介绍并分析了系统各模块，用户对系统功能的需求，以及对数据库，数据表的设计。本文的系统包含电影展示、评分模块、用户信息、推荐结果等模块。

本系统采用改进之后的算法来改进相似度计算上的某些方面。用户可以在使用推荐电影系统时根据几种推荐算法更加合理的选择推荐方式。经过改进过的几种算法使得用户的可选择性更高，有多种角度可供参考并进行推荐。利用 css 语言美化了网页，结合前后端框架，利用内置的功能实现了用户登录注册等基本功能，与推荐结果的前端展示。使系统结构更为更加整洁。

本文在整个系统架构中有涉及到的编程语言包含 Python，Html5，JQuery，CSS，MySQL。用到的是 Django 框架，通过 Django 连接前后端。本文还分析了系统的需求，并设计根据需求来运行系统。

**关键词：**电影推荐系统；交替最小二乘算法；基于邻域推荐；个性化服务

# Movie recommendation system based on users and items

**Abstract:** This paper designs a movie recommendation system. There are more and more movie resources on the network, which are more and more complex. So it is necessary for us to design an efficient personalized movie recommendation system in order to help people solve the problem of movie search. Therefore, this paper designs a movie recommendation system to provide each user with movie resources that meet their interests.

The system plans to use the dataset crawled by the network. First, all the files contained in the data set are stored in the database and divided into test set and training set, and then the movie recommendation list is generated by analysis. This paper also introduces some basic recommendation algorithm principles and basic formulas. The system also analyzes and explains the structure and functions of the recommendation system. This paper introduces and analyzes the modules of the system, the user's requirements for the system functions, as well as the design of database and data table. This system includes film display, scoring module, user information, recommendation results and other modules.

This system uses the improved algorithm to improve some aspects of similarity calculation. Users can choose more reasonable recommendation methods according to several recommendation algorithms when using the recommendation movie system. The improved algorithms make users more selective, and there are many angles for reference and recommendation. Using CSS language to beautify the web page, combining the front and back end framework, using the built-in functions to achieve the basic functions such as user login and registration, and the front-end display of recommendation results. Make the system structure cleaner.

**Keywords:** Movie recommendation system; Collaborative Filtering; criterion; Based on neighborhood recommendation; Personalized service

# 1 绪论

## 1.1 研究背景及意义

假如我们想帮用我们亲爱的用户找寻他喜欢的物品，真的十分困难。信息过于杂碎，远远超过了人们可以手动遍历的范围。就比如我们逛宜家，不可能把整个商场搬空，不可能一一遍历。面临大量物品的挑选我们也会时不时的点开购物软件，面对太多太多的大大小的琐碎商品不知道要怎么搞。经济学中有一个专业词汇叫长尾理论。这套东西在互联网领域而言，就是讲的是最具热度最红的商品人人想要，不出名丝毫没有热度的商品白送都没有人关注。这种现象当然造成了资源上的极大的浪费，同时使得兴趣爱好小众的用户得不到想要的商品。当代互联网的这个爆炸当量的信息量，假使我们将所有内容都集中在用户网站首页，用户是看不过来，也没法子阅读的，用户对于有效信息的利用率就会十分低下，没有效率。那么我们就迫切的需要有一个系统为我们过滤掉信息，好的推荐系统能让用户三番五次地访问他感兴趣所在的页面，并且用户总是能够有效找到想要的推荐信息，系统如果成功推荐了一个被用户证明其确实是其感兴趣的内容后，我们对就更能把握这个用户的兴趣形象。最后我们确认能够找到用户精确的兴趣所在的时候，我们此时就可以根据这个来为用户量身定制个性化的系统，使得我们构建的平台可以满足大量不同用户的需求。到底啥是推荐算法？我们倾向于把这个概念简化为一个有输入输出的函数。我们把用户和 `item` 的各种属性和特征当做算法的输入参数，返回一个按照用户喜好度排序的物品集合，经过推荐算法各种处理后，推荐给用户。

推荐系统在日常生活中与我们基本上是形影不离息息相关，大众层面上比较红火的市面电子商务应用都基本用到了推荐系统，主要就是帮助人们发现感兴趣的物品，削减繁杂的不必要的内容。想赚钱的广告商还会利用推荐系统来查看感兴趣的用户，有针对性的投放广告。举一个亚马逊较著名的例子，假如你在亚马逊购置了一本书，付款完毕以后，它会时不时的向你的手机，邮箱，相关社交软件发送令人恼火的推销信息，此外还有诸多的电影和视频网站，像美国最著名的也是全球影响力最大视频网站脸书。国内的个性化音乐平台 qq 音乐，豆瓣电台。社交网络 Facebook 等。

找不到用户真正要的信息问题，产生了搜索引擎，但是引擎会失效如果用户无法准确表达提供的信息关键字。就在这样的复杂背景下推荐系统的出现为的就是解决上述的问题，将用户

与信息紧密相连，让用户不费力气就能找到信息，为了防止信息变得冗杂并且让信息能够展现在对他感兴趣的人群中，进而实现彼此互相成就，因此考虑引入推荐算法，很多网站可以应用个性化推荐技术，比如视频、音乐网站、购物网站等。推荐算法被应用到各个领域，最著名的使用推荐算法来推荐视频和音乐商品的服务网站有网飞，油管，声破天，亚马逊，还有为用户推荐社交信息的媒体平台脸书和推特甚至还有餐厅和线上交友。急切需求一种向用户自动推荐的方案，并且这个系统是基于用户的 **interest**。但最初的自动推荐系统，只会将当下热门的商品推荐，并没做出一个符合用户喜好的系统，因此本文考虑做出一个根据用户和物品来推荐的系统。

推荐系统的流行就是为了解决物品和信息越来越繁杂，种类与信息量令人眼花缭乱，人们没有时间也没有那个精力去一一阅读了解，当然，也不可能了解的完。现代信息化社会普罗大众就更要以一个有效率的方法去筛选需了解的事物和商品，在茫茫的物品和信息之间，架起一支小舟，联系着这些信息物品和用户，它就像是一个专门量身打造的知音导购，帮助人们更好、更快地选择自己感兴趣的、自己需要的东西。

在推荐算法的加冕下，仅仅耗时短短几年，各色推荐网站用户的增长速度与上线时间就令人惊叹，受到了市场的热烈的追捧和高估值。

现阶段各种海量数据多如牛毛，相比于过去的信息缺乏，现在的数据量扩大到了一个庞大的局面，怎样判断衡量一个系统是否颇具效率呢？那就是对于庞大的数据，系统是如何做到筛选的。想要具有良好的用户体验，系统会将用户最想要的信息展现给用户，筛选过滤掉无用信息。节省了用户在不需要的数据里晕头转向。

推荐系统通过各种各样的手段收集你的信息，有些购物网站就会收集你的数据来计算评测你的兴趣。通过各式各样的商品让你打分，他会鸡贼的偷偷记录你的行为数据，浏览界面，哪里点开了，点进了几秒，你反复浏览了那些商品？你购物车里有啥？你最终买了啥？最终拿你日常浏览路径弹出相关符合你兴趣的商品，再给你建立一个长期文件袋保存你的数据。

好多互联网巨型企业都会收集你的诸多数据，你的每一步浏览记录都将被记录在册，往后用来给你打广告，甚至不想透露姓名的 **buyer** 也会记录你的数据和你个人信息相关。你的所有网上记录都会被人盯着记录。当然会导致互联网企业的杀熟客，老用户的价格比新用户高，你喜欢听的不收费的歌，当你成为老用户就开始收 **VIP** 的钱。



有些算法也会给出错漏百出的推荐，这些推荐不仅会失去用户的信任，还于销售无用。举例说明每个去购物的人都喜欢买麦片，那就不应该向见到的每个客户推荐麦片。不仅于销售无助，还遭用户的烦。所以就要制定规则不推荐麦片。

还有不要向用户售卖容易亏本的产品，或者减价推荐卖不出去的商品。

想要获得用户的信赖也是非常不容易的。一个极力宰人的商店显然获取不了用户的信赖。这是就需要向用户公开你的一部分，给用户一个大体的认知，并允许用户相对的选择推荐给他自己的商品信息，假如他不喜欢他可以拒绝。并给推荐给他的商品加上一定贴心的合情合理的说明。给出一定的附加信息能使得用户明确推荐是否有效。

推荐算法在商业领域的应用极大带动了销售的增加，这也是推荐系统的主要应用。因此评价一个推荐算法的好坏，就要看他 and 实际的差距。大多互联网公司都会看原先预测这个用户对某项产品的打分与他的实际打分之间的差距来评测推荐算法的效益。同样的是原先的推荐和用户真实买入的商品差距有多大。但是这样会将用户自己找的商品计算为推荐算法带来的。惊喜和多样化也是推荐算法可以为用户带来的东西。

惊喜可能会导致少见的推荐结果，假使一件商品对大部分用户来说无关紧要，只对小部分用户有价值。虽然不太常见，但是一旦推荐给了恰当的用户，就会给用户带来意外之喜，提高用户的满意度。就算再喜欢一类商品，看到推荐列表里面全是这些商品，也会感到厌倦。给用户推荐全方面都兼顾的商品，才是较为成熟的推荐系统。仍然有大量问题需要解决，推荐算法可能需要给用户推荐一些新的东西，新的想法，防止人们对此疲倦。随着推荐量增加，与收集数量的增加，推荐效能也会变得更佳。

推荐领域的不断发展，对于个人而言被认为是一件有利于世界范围内的信息交流，加快技术发展的事，加快了我们认知新事物新技术的速度。假使个性化推荐系统能将我们最感兴趣的行业最新知识推荐给我们，那么这是消除隔阂增加信息的价值的非常好的手段需要我们附以心血研究。这是一件对人类和世界来说非常有价值的事情。

## 1.2 国内外研究现状

推荐系统自问世以来，就被大量应用于生活场景中，如今随着发展，AI 与 DM 等领域的研究更加推动了推荐系统的的发展，大量的推荐技术被用在诸如购物网站，流行音乐网站，电影网站中，而本文使用利用用户之间的相似性来实现的算法是推荐用户感兴趣的电影给用户，但

有两个相对比较难解决的问题，一是可扩展性，随着用户和资源的增多会使系统性能变得越来越低，另一个是稀疏性由系统初期得到的评价反馈不足引起。就近年来行业的发展纵向来看的话，推荐算法发展问题第一是，考虑附加上更多的信息来缓解 user-item 矩阵的稀疏问题，就比如大众的评论信息或者上下文语境信息等，其次就是利用 CNN 来处理文本方面的信息，利用深度学习技术。提高推荐列表的可解释性，比如利用脸书的朋友信息来解释。惊喜也是需要考虑到的热点问题，用户想要探知未知领域的渴望。利用哈希技术来进一步提高效率，再者推荐隐私问题值得考虑，如何做到在保护用户隐私问题同时缓解数据稀疏。

从推荐算法的发展历程来看，从协同过滤到矩阵分解再到隐含因子分解，推荐系统的功能可以划分成新方法的引用，和调整过去的算法并改进来提升计算效率和应用层面，这也带来了进一步的更新。几个热点包括了有人工智能的方面的介入带来的深度学习技术的应用。成功带来了在诸多领域的巨大发展，把深度学习与推荐系统的结合更是新兴话题，因为其提取特征与表达能效空前强大。主要带来的应用有深度学习展宽线性与非线性模型的深层结构，加强了表达能力。提升了传统编码方式的联系广度，更加体现了各方面之间的各种角度的联系，引入更多的非线性单元。自动学习高级交互模式，补充由于手动来的缺陷。

还有一些问题诸如，在一段时间内要处理的数据量过于庞大，怎样更好的在体量较为庞大的平台上应用也是技术问题之一，还有多种类数据的处理，将大量不同种类的数据融合至一个模型中使之发挥作用，再其次就是用户的长期兴趣与短期兴趣相结合，对此进行推荐。

推荐场景中，强化学习也是一大应用。强化学习的回报机制与交互相关行为适合应用在诸多新闻平台，商务平台中。提高系统的解释度，也有便于提高系统的效率和对用户的解释和交流。举例，能否以用户容易接受的方式来拉近与用户之间的距离，更容易和用户的沟通，能够提升系统系统透明度，用户的心里满意程度和信赖。通过添加一定说明信息引起用户的关心。同时，不止给用户只推荐特定死信息，只看重准确度少了灵活。不能用户喜欢吃西瓜就只给他推荐西瓜，用户可能还喜欢草莓等等，假使只给用户推荐西瓜就会给用户带来不好陈旧的使用体验。推荐的公平程度也是要慎重考量的问题，只要推荐多样化提升，公平程度也会提升。关于推荐系统的提升，还有很多能做到的方面，诸如将经济学要素应用进来，解决更多常见方面的问题。运用新的深度学习方法解决过去难处理的问题取得了一定进展。

推荐系统设计的时候，不可避免的问题就是用户的隐私问题。用户的过去浏览历史与个人隐私信息等等，这些信息的保护也是会带来巨大风险严重方面。

因为机器学习技术日趋成熟，面对复杂问题有更多处理方式，各大网站也在考虑采用机器学习来重新构建系统，传统推荐算法确实在被淘汰，传统方法没办法再在更精一步的推荐，效果上相对于机器学习也有不少的劣势。再加上商业上的巨大维度特征和复杂庞大的模型，算法框架与基础也得到提升，同时还要考虑到多样性与准确率与用户的反馈并重，更好的服务用户。

### 1.3 论文主要工作

本文主要研究内容包括：

研究传统的推荐算法，对传统的推荐算法进行一定改进，完整设计实现出一个电影推荐系统。

### 1.4 论文主题结构

本文的安排如下所示：

#### 第一章 绪论

绪论部分主要内容如下图 1-1 所示：

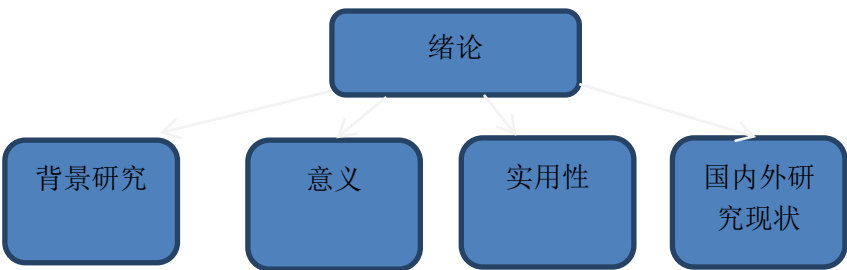


图 1-1 绪论

#### 第二章 有关推荐算法的研究

介绍几种常见熟知的典型推荐算法和相似度计算方式，同时介绍一些常用的指标测评方式。

#### 第三章 推荐系统的实现技术研究

介绍有关实验方面涉及到的技术以及大体上的设计步骤。

#### 第四章 推荐系统的设计与实现结果

分析用户对于个性推荐的相关需求以及系统各个模块的功能，数据库方面的设计，前

端的相关架构等。

## 第五章 总结与展望

总结了本系统实现的大体实现过程，简单描述了不足与日后加以改进的地方。

## 2 关于推荐算法的研究

### 2.1 有关推荐算法的相关简介

#### 2.1.1 协同过滤算法

首先，所有的推荐算法简而言之都只有一个最终目的，就是为了给用户推荐商品，什么商品？当然是他最可能购置的商品，进而提高销售量和转化。一般都是相同数量物品和用户的数据，只有少部分的用户与数据之间存在确实一定存在的评分数据，其余没有评分数据，为了将最高评分的物品推荐给用户，此时就用已有稀疏数据来预测评分关系。包括了在线协同，离线过滤这两个方面。在线协同即为找到物品的相关在线数据，这些物品必须是用户可能会喜欢的并滤掉用户不感兴趣的物品数据。

就一般情况来说，协同过滤有下列几种。

基于用户的协同过滤主要把相似度作为评价准则，比如有甲一和乙二两个用户，他俩相似，那么我们就给甲一推荐乙二可能会喜欢的东西。想找到并推荐用户评分最高的那些电影给用户，那我们就要找出了相似用户他们喜欢的那些电影并对预测目标用户进行对应打星。就拿微博的推荐好友机制来举例说明，我和戊不是互关，但我的互关人里面有一些人关注了戊，那么新浪就会把戊推荐给我作为我的新的关注，同样情况我和丙不是好友，但我的朋友里面有一些人和丙是朋友，那么新浪就会把丙推荐给我作为我的新的好友。实际中来说，假如我关注的人里面有很多人也关注了丙，事实上就一定有关丙的人有不少我的好友。这就是基于相似 user 的推荐。

基于商品的协同过滤和基于用户的有一定的相似，即物与物，不过基于物品就要找到物品之间的相似度，就比如，很多人看过一部番剧 A，他们都同样喜欢 B，那么就假定 A 与 B 大体上是相似的物品，就这样衡量物品之间的相似度，还有一个有趣的假设，就是假设甲一用户的兴趣只有特定局限的几个，那么假如 P 物品和 H 物品都属于甲一用户的兴趣行列，那么我们假定 P, H 就隶属于这有限的几个领域，假使 P, H 属于很多用户的兴趣行列，那么就假设 P, H 属于同一领域，从而拥有极大的相似度。一旦找到用户对特定商品的评分，那么我们就可以将最高评分的相似物推荐给用户，采取预测相似度极高的类似物品的方式。

假使我们在相似度关系比对中进行，如果拿复杂度计算作为考量标准，基于用户的肯定比

基于物品的高。而基于物品方面的协同过滤方式，相似度短期难以发生太大的变动，可进行相对容易的离线计算，准确度较高。但推荐的多样性较为低下。

以上两种都是基于邻域的算法。

### 2.1.2 基于内容的推荐算法

基于内容的推荐算法主要来自于对于信息做出的简单处理，就是根据用户以前被记录过的行为或者物品自身的相关方面向用户推荐以前没接触过的推荐项。特别适合用于文本信息相关方面，诸如新闻。原理是用户非常喜欢与自己感兴趣的东西相似的内容，比如你看了纳尼亚传奇 1，基于内容的算法经过计算觉得纳尼亚传奇 2 与你喜欢的内容存在相似关联性，那么他就会把后一本书放在你的推荐首页。这处理了冷启动的相关问题，不会出现没出现过的物品不去推荐的问题，缺点则是可能会推荐物品相重。比如新闻反复推荐你看过的内容，推荐听过好几遍的歌让你烦不胜烦。可以采取手动打标签，来提取内容。

凭借经验定义计算公式，再根据公式计算来自证抽取一些特征从每个物品中，为了了解喜好特征，我们利用一个用户过去的特征数据来比较用户的特征和物品的特征，推荐与用户过往兴趣较为相似的物品给到用户。

即用户购买过的物品，在找出这些物品根据属性具备原则查找类似物品，为用户推荐。将 G 产品分解为一系列的标签，诸如一个水杯，可以分为颜色，材质，生产地等等等，基于用户的浏览删除收藏来考量每个用户的专属标签，对于每个产品计算余弦相似度。一般包括了三步走，为每个物品抽取一点深层内容来代表此物。利用一个用户过去购买过的商品的特征数据，来学习这个用户的爱好特征。再根据比较上一步得到的特征，推荐一组相关性最大的商品。

### 2.1.3 基于标签的推荐算法

首先打标签也是很常见的行为。用户可以根据标签迅速找到想要的物品，同时网站也可以根据用户打出的标签更好的推荐物品。标签是一种用来描述信息内涵特征的关键词，大多数网站均有应用。

## 2.2 实现系统的算法介绍

### 2.2.1 基于用户的算法

首先我们假定存在两个用户，分别是  $q$  和  $p$ ，用  $N(q)$  这个电影列表表示用户  $q$  评价过的所有电影， $N(p)$  电影集合代表用户  $p$  过去打过分的电影。计算相似度考虑采取余弦相似度：

$$W_{qp} = \frac{|N(q) \cap N(p)|}{\sqrt{|N(q)| |N(p)|}} \quad (2-1)$$

由于上式计算较为不准确，因此考虑改为采取下列方式计算相似度：

$$W_{uv} = \frac{\sum_{i \in N(u) \cap N(v)} \frac{1}{\log(1 + |N(i)|)}}{\sqrt{|N(u)| |N(v)|}} \quad (2-2)$$

UserCF-IIF 式是改进过的基于用户的算法记为式（2-2）通过  $\frac{1}{\log(1 + |N(i)|)}$  此式惩罚用户的

兴趣集合。计算得到用户之间的相似度后，推荐算法根据推荐结果推荐与他兴趣最相似的  $K$  个用户喜欢的电影给用户，下列公式是对于用户对电影感兴趣度的计算。

$$P_{ui} = \sum_{v \in I(i) \cap S(u, k)} W_{uv} r_{vi} \quad (2-3)$$

采取 Spark ALS 算法改进，即交替最小二乘法，由于用户会对商品做出评价，但是不会每次都给每个物品打分，因此考虑采用 ALS 算法补齐未打分的部分并进行分预测。把用户和商品的评分表分为两个矩阵的相乘，采取中间一部分的隐藏信息作为中间信息进行补充。就如喜欢芬达也喜欢雪碧，就可以用喜欢小苏打水作为隐藏的因子，可以分成产品蕴含隐藏因子的程度与用户对隐藏因子的偏好程度相乘。这两项互相耦合，因而可以转化计算最小二乘，初始值随意设定，根据第一项计算第二项，在根据第二项推下一个第一项，就这样反复进行迭代，直至收敛。

### 2.2.2 基于物品的算法

物与物之间的相似度计算，通过计算物物之间相似程度来考虑用户兴趣进而推荐给用户可能喜欢感兴趣的物品。

$$W_{ij} = \frac{|N(i) \cap N(j)|}{\sqrt{|N(i)| |N(j)|}} \quad (2-4)$$

式（2-4）为计算物品相似度的公式，它惩罚了物品  $j$  的权重，运用到本系统中可以有效避免某一热门电影和很多其他多数电影都产生较大的相似度

在基于物品算法中，每个用户对物品的贡献都不相同，活跃用户对物品相似度的贡献更大，假如很多用户都喜欢 A 物品也喜欢 B 物品，那么可以近似认为 A 与 B 物品之间存在部分相似。所以考虑采用下列经过改动的计算式来计算相似度：

$$W_{ij} = \frac{\sum_{u \in N(i) \cap N(j)} \frac{1}{\log(1 + |N(u)|)}}{\sqrt{|N(i)| |N(j)|}} \quad (2-5)$$

式（2-5）中的  $\frac{1}{\log(1 + |N(u)|)}$  处分了活跃用户对物品相似度的贡献。之后通过下列公式计

算来用户与物品之间兴趣度。

$$P_{vj} = \sum_{i \in I(v) \cap S(j,k)} W_{ji} r_{vi} \quad (2-6)$$

采取 ALS 算法改进部分和 2.2.1 节类似

### 2.3 相似度计算

下列是主要几个计算相似的的典型方式：

#### （1）余弦相似度

想要衡量两个向量的相似度就用两个向量之间的夹角的余弦值大小表示。假设  $n$  维空间中存在两个向量如公式所示，式（2-7）为计算其余弦相似度的公式。

$$Sim(i, j) = \cos(\vec{i} \cdot \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{\|\vec{i}\| * \|\vec{j}\|} \quad (2-7)$$

#### （2）欧式距离

欧式距离是指任意两个向量之间的自然长度。同样的，假设存在两个向量，欧式距离表达式如式（2-8）所示。

$$Dist(M, N) = \sqrt{\sum_{i=1}^n (m_i - n_i)^2} \quad (2-8)$$

#### （3）皮尔逊(Pearson)相关系数

这个算法也是考察变量之间相似度的计算，这个算法是相当于在余弦相似度算法上做出的一些改动。因为余弦相似度的维度相对缺失，所以考虑进行改进。它在计算余弦相似度的每个向量之前都要进行中心化，即在计算所有元素的均值后再用向量的维度值减均值得出的。这个就是中心化的相关处理。

有两个向量，这两个向量间的皮尔森相关系数计算公式如式（2-9）所示。

$$\rho_{J,K} = \frac{\sum (J - \bar{J})(K - \bar{K})}{\sqrt{\sum (J - \bar{J})^2 \sum (K - \bar{K})^2}} \quad (2-9)$$

#### 曼哈顿距离

明显是一个非负的数，描述两个点的距离，最小就是两个点相重值为 0。基本与欧式距离的

意思相似，只要两个点的横纵坐标分别相减取绝对值再相加。此式由于只要加减，使得损失误差较小，也更容易做计算。

$$Sim(x, y) = |x_1 - x_2| + |y_1 - y_2| \quad (2-10)$$

## 2.4 如何测评推荐算法

### 2.4.1 评分预测

(1) 公式如下：

$$MAE = \frac{1}{|T|} \sum_{u,i \in T} (r_{ui} - \hat{r}_{ui})^2 \quad (2-12)$$

就是用真实值减去预测值再平方求和平均，也就是在测试集上测评的损失

RMSE，就是均方误差用来更好的描述数据，是回归算法评价指标，是 MSE 开根号：

$$RMSE = \sqrt{\frac{1}{|T|} \sum_{u,i \in T} (r_{ui} - \hat{r}_{ui})^2} \quad (2-11)$$

式 (2-11) 中的 T 表示总数据量， $U_{ir}$  表示的是评分， $\hat{r}_{ui}$  表示的是预测评分。

假如做价格评测，每单位内的价格是万元级别，那么相减的平方数量单位会非常的大，此时开根号就能使得结果与数据的级别相同。

### 2.4.2 TopN 推荐

就是使用一个特定大小的数组，用来存放最高级别的特定长度对象，越高级的对象在数组中的序号就越小，就是要找到最大的特定长度的数据从已知数组中。这种预测准确率一般是通过计算以下两种方式得到。

准确率计算公式：

$$Precision = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |T(u)|} \quad (2-13)$$

召回率计算公式：

$$Recall = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |R(u)|} \quad (2-14)$$



### 3 推荐系统的实现技术研究

#### 3.1 实验环境介绍

实验环境如下表 3-1 所示:

表 3-1 实验环境

处理器	Intel(R) Core(TM) i5-7200U 2.5GHz
内存	4.00GB
运行环境	Windows 10
软件	Sublime Text3, MYSQL, Pycharm
编程语言	Python

#### 3.2 系统实现相关技术介绍

实验所需技术如表 3-2 所示:

表 3-2 系统所需技术

技术类型	技术名称
推荐算法开发语言	Python3
前端开发语言	Html5、CSS3、Jquery
web 框架	Django
数据库	MySQL

#### 3.3 系统的设计

本系统主要由注册, 登录, 注销, 评分, 查看结果部分组成。将数据集进行处理, 并将所得结果放入数据库中, 并使用框架和 css 进行前端页面展示和架构。

##### 3.3.1 数据库设计

数据库设计如表 3-3, 3-4 所示:

表 3-3 数据库设计表


名	类型	长度	小数点	允许空值 (Null)	
id	int	10	0	<input type="checkbox"/>	 1
uid	int	11	0	<input checked="" type="checkbox"/>	
keyword	varchar	100	0	<input checked="" type="checkbox"/>	
img_url	varchar	255	0	<input checked="" type="checkbox"/>	
art_url	varchar	255	0	<input checked="" type="checkbox"/>	
img_key	varchar	255	0	<input type="checkbox"/>	 2
ct	timestamp	0	0	<input checked="" type="checkbox"/>	

表 3-4 数据库设计表

名	类型	长度	小数点	允许空值 (	
id	bigint	20	0	<input type="checkbox"/>	 1
login_name	varchar	50	0	<input type="checkbox"/>	
login_password	varchar	50	0	<input type="checkbox"/>	
last_login_time	timestamp	0	0	<input type="checkbox"/>	
last_login_ip	varchar	50	0	<input checked="" type="checkbox"/>	
login_count	int	11	0	<input checked="" type="checkbox"/>	
is_enable	int	11	0	<input checked="" type="checkbox"/>	
sex	int	4	0	<input checked="" type="checkbox"/>	
remark	varchar	255	0	<input checked="" type="checkbox"/>	
add_time	timestamp	0	0	<input type="checkbox"/>	

3.4 电影的数据采集系统的实现

该系统根据系统的要求设计，包括各种功能模块和数据库的设计，然后通过技术手段实现系统。

网络爬虫的流程图如下图 3-1 所示:

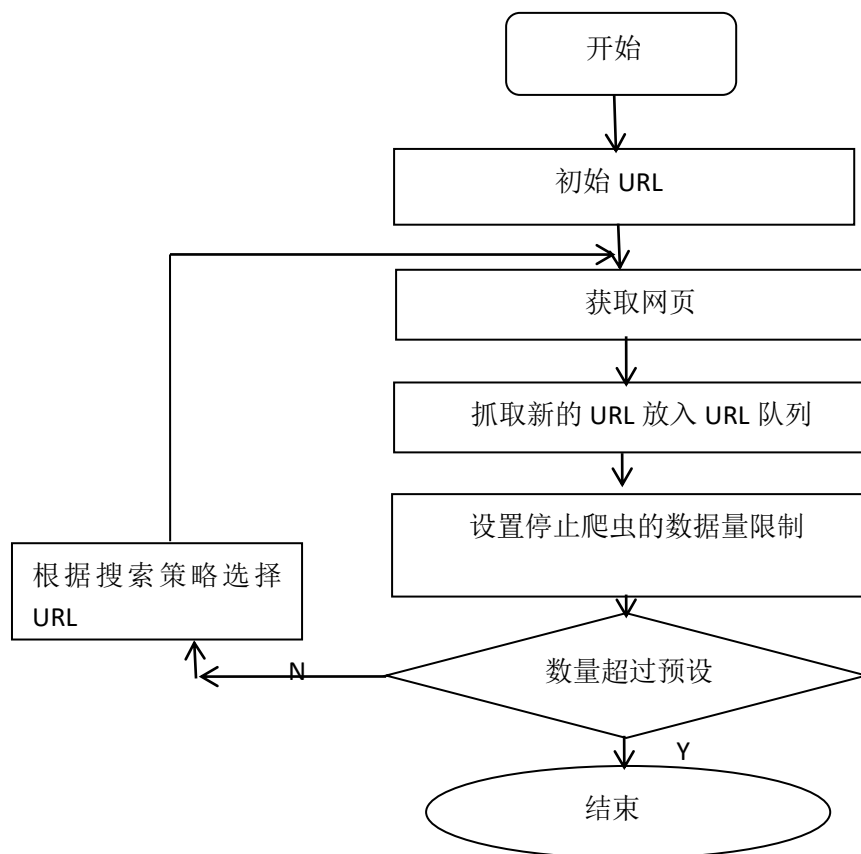


图 3-1 爬虫流程图

### 3.5 URL 构造策略

网络爬虫主要内容就是将网上的部分内容储备到本地形成备份。我们根据网页的异同点设计自动爬网 URL。在编写网络爬虫的过程中，尝试用 URL 来发起请求首先要找到一个可以调用的 URL 来获取服务器的资源。尝试拿到网页的各种响应再来根据此提取我们想获取的信息。信息时代爬虫就是用来获取并且处理信息的，信息的储备非常重要，关系未来的发展，搜索引擎就可以类比爬虫，也就是对信息的各种收集。由于广义信息爬虫并不能满足人们的广泛需求，从而诞生了自己来下定义的爬虫，爬取范围更广可以不遵循协议爬取信息。网站末尾的数字形式的尾巴是网页的标记。爬网的维度用初始变量来控制。

### 3.6 数据提取与存储采集

为了方便地提取数据，我们找到了需要被爬行的目标电影网站，根据 F12 打开调试窗口，参见图 3-1，然后锁定要提取的类，并将其定位在 XPath 中。

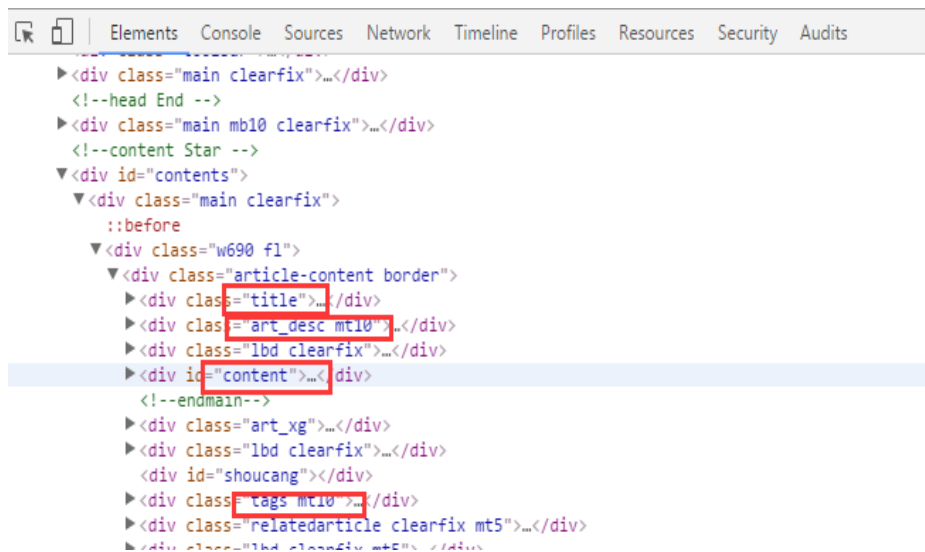


图 3-2 目标网站网页图

## 4 推荐系统的设计与实现结果

### 4.1 数据爬虫模块的实现

部分代码如下图 4-1,4-2 所示：

```
base_url = 'https://movie.douban.com/top250'
results = {}
new_url = 'https://www.douban.com/doulist/30299/?start=0&sort=seq&playable=0&sub_type='
ids = 0
import csv

tasks = []

class Movie:
    def __init__(self, id, title, description, star, leader, tags, years, country, director_description, image_link):
        self.id = id
        self.star = star
        self.description = description
        self.title = title
        self.leader = leader
        self.tags = tags
        self.years = years
        self.country = country
        self.director_description = director_description
        self.image_link = image_link

    async def fetch(url):
        async with aiohttp.ClientSession() as session:
            async with session.get(url) as response:
```

图 4-1 部分实现代码

```

    async with session.get(uri) as response:
        print(response.status)
        assert response.status == 200
        return await response.text()

async def write_images(image_link, image_name):
    print('write images....', image_link)
    async with aiohttp.ClientSession() as session:
        async with session.get(image_link) as response:
            assert response.status == 200
            with open('movie_images/' + image_name + '.png', 'wb') as opener:
                while True:
                    chunk = await response.content.read(1024)
                    if not chunk:
                        break
                    opener.write(chunk)

async def parse_list(html):
    soup = BeautifulSoup(html, 'html.parser')
    movies = soup.find_all('div', {'class': 'doulist-item'})
    movie_list = []

    for movie in movies:
        try:
            title = movie.find('div', {'class': 'title'}).text.strip().replace('/', ' ')

```

图 4-2 部分实现代码

## 4.2 需求分析

人们都希望可以高效的找到自己感兴趣的电影并以此节省时间。通过数据集里面大量评分数据以此计算。

## 4.3 用户需求模块

如图 4-3 是系统中功能图:

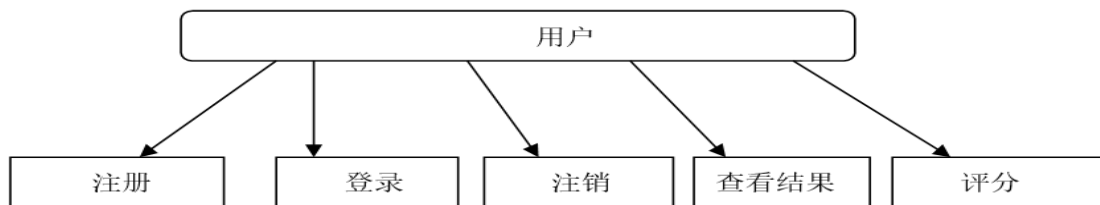


图 4-3 用户图

## 4.4 系统实现

实现本系统主要需要以下几种编程语言:

Python 用作编写后端脚本和算法实现模块, 将项目连接上数据库并将生成的用户数据存入, 并在前端页面展示生成的推荐列表。Html5 用作制作前端页面。Css3 用来美化并处理前端页面。

## 4.5 系统功能模块简述

### 4.5.1 数据初始处理

本文设计的推荐系统对爬取得到的数据集里面的 csv 文件进行处理, 只保留三个有效信息段存入新的表中并放入数据库。

4.5.2 注册登录模块

系统两大主要应用模块如图 4-4 所示:

用户注册

Username:

xueqiubei

Email:

Password:

\*\*\*\*\*

Password Confirmation:

name:

Phone:

Address:

✕

✓

用户登录

Username:

xueqiubei

Password:

\*\*\*\*\*

✕

✓

4-4 登录与注册界面

用户登录的核心代码如下:

```
<h3>登录</h3>
<form class="form" action="{% url 'login' %}" method="post">
  {% csrf_token %}
  {{ form.non_field_errors }}
  {% for field in form %}
    {{ field.label_tag }}
    {{ field }}
    {{ field.errors }}
    {% if field.help_text %}
      <p class="help text-small text-muted">{{ field.help_text|safe }}</p>
    {% endif %}
  {% endfor %}
  <button type="submit" class="btn btn-primary btn-block">登录</button>
  <input type="hidden" name="next" value="{{ next }}" />
</form>
<div class="flex-left top-gap text-small">
  <div class="unit-2-3"><span>没有账号? <a href="{% url 'users:register' %}">立即注册</a></span></div>
  <div class="unit-1-3 flex-right"><span><a href="reset_password.html">忘记密码? </a></span></div>#</div>
</div>
```

注册的核心代码如下:

```
<meta charset="utf-8">
<meta http-equiv="x-ua-compatible" content="ie=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
<title>注册</title>
<link rel="stylesheet" href="https://unpkg.com/mobi.css/dist/mobi.min.css">
<style>
.errorlist {
  color: red;
}
</style>
</head>
<body>
<div class="flex-center">
<div class="container">
<div class="flex-center">
<div class="unit-1-2 unit-1-on-mobile">
<h3>注册</h3>
<form class="form" action="{% url 'users:register' %}" method="post">
  {% csrf_token %}
  {% for field in form %}
    {{ field.label_tag }}
    {{ field }}
    {{ field.errors }}
    {% if field.help_text %}
      <p class="help text-small text-muted">{{ field.help_text|safe }}</p>
    {% endif %}
  {% endfor %}
</form>
</div>
</div>
</div>
</div>
</body>
</html>
```

4.5.3 电影分类模块

系统分类出不同的实现区域，如图 4-5，4-6 所示：

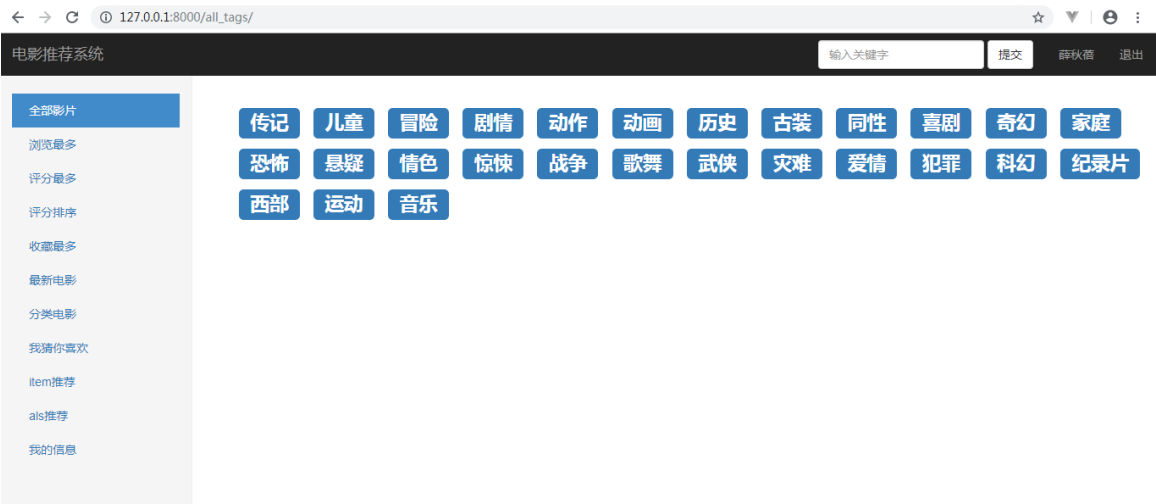


图 4-5 系统分类界面

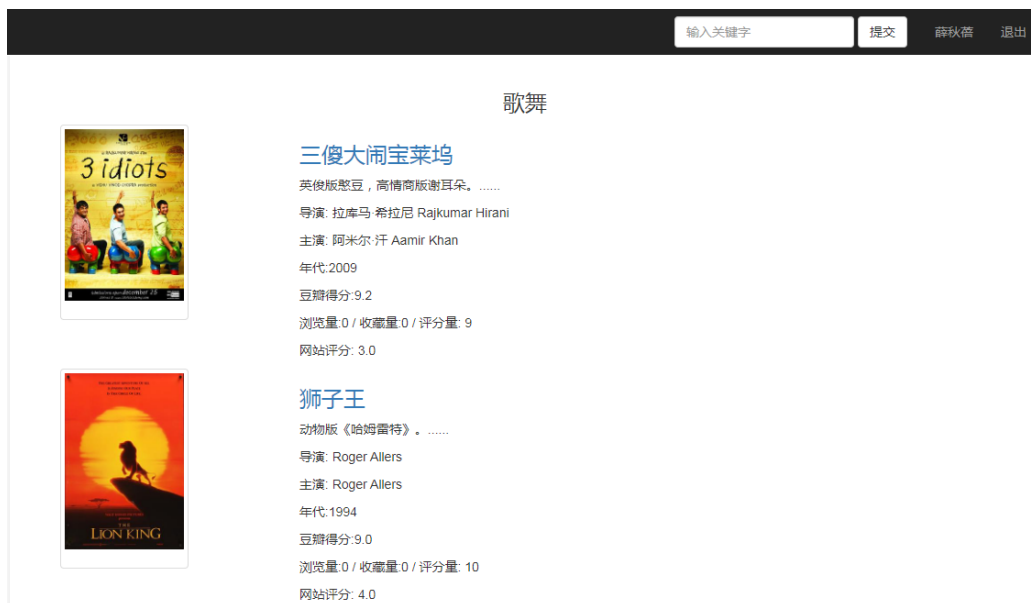


图 4-6 歌舞片分类

分类实现代码如下：

```
{#
    </p>#}

<div>
  <hr />
  <ul class="nav nav-tabs nav-justified" id="index">
    <li class="active"><a href="#tab0" data-toggle="tab">动作</a></li>
    <li><a href="#tab1" data-toggle="tab">恐怖</a></li>
    <li><a href="#tab2" data-toggle="tab">喜剧</a></li>
    <li><a href="#tab3" data-toggle="tab">动画</a></li>
    <li><a href="#tab4" data-toggle="tab">科幻</a></li>
    <li><a href="#tab5" data-toggle="tab">犯罪</a></li>
    <li><a href="#tab6" data-toggle="tab">爱情</a></li>
    <li><a href="#tab7" data-toggle="tab">剧情</a></li>
  </ul>
</div>

{#动作片#}
<div class="tab-content" id="my-content">
  <div class="tab-pane fade in active" id="tab0">
    <ul class="figures_lists">
      <li class="list_item" data-trigger-class="list_item_hover">
        <a _boss="film" target="_blank" class="figure" tabindex="-1">
          
        </a>
      </li>
    </ul>
  </div>
</div>
```

#### 4.5.4 用户评分反馈模块

如图 4-7 所示：





图 4-7 用户对歌舞片的评分

#### 4.4.5 用户评分记录模块

客户登录进系统后进入自己记录过的账户，就会将过去历史评价记录显示出来。如图 4-8 所示:

电影名	评分	时间	删除
战争之王	3.0	2020年5月12日 21:36	<a href="#">确定</a>
锈与骨 De rouille et d'os	4.0	2020年5月12日 21:36	<a href="#">确定</a>
倾城之泪	4.0	2020年5月12日 21:36	<a href="#">确定</a>

图 4-8 用户看过并评价的电影集合

#### 4.4.6 推荐算法流程

这是整个系统重要部分，系统记录下用户的码号每当用户登录系统并做出评分，如果用户再登入系统时，就会将初始的相关信息和再评分重新计算考虑并作为结果展示。

推荐的相关步骤如图 4-9 所示:

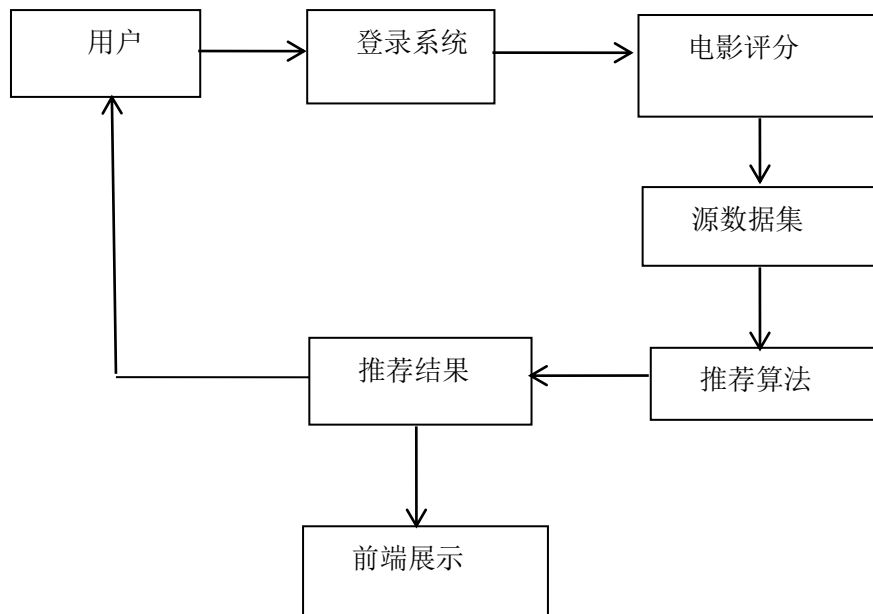


图 4-9 系统推荐流程图

#### 4.4.7 显示推荐模块

根据登录用户对电影的评分反馈，给出图 4-10，4-11, 4-12 结果：

提交
薛秋笛
退出

### 依据item推荐




#### 肖申克的救赎

希望让人自由。.....

导演: 弗兰克·德拉邦特 Frank Darabont

主演: 蒂姆·罗宾斯 Tim Robbins

年代: 1994

豆瓣评分: 9.7

浏览量: 0 / 收藏量: 0 / 评分量: 11

网站评分: 4.0

#### 霸王别姬

风华绝代。.....

导演: 陈凯歌 Kaige Chen

主演: 张国荣 Leslie Cheung

年代: 1993

豆瓣评分: 9.6

浏览量: 0 / 收藏量: 0 / 评分量: 11

网站评分: 1.0

图 4-10 基于物品的推荐结果

输入关键字

提交

薛秋薇

退

依据item推荐



### 海上钢琴师

每个人都要走一条自己坚定了的路，就算是粉身碎骨。.....

导演: 朱塞佩 托纳多雷 Giuseppe Tornatore

主演: 蒂姆 罗斯 Tim Roth

年代: 1998

豆瓣评分: 9.3

浏览量: 0 / 收藏量: 0 / 评分量: 10

网站评分: 2.0



### 肖申克的救赎

希望让人自由。.....

导演: 弗兰克 德拉邦特 Frank Darabont

主演: 蒂姆 罗宾斯 Tim Robbins

年代: 1994

豆瓣评分: 9.7

浏览量: 0 / 收藏量: 0 / 评分量: 11

网站评分: 4.0


表 4-11 基于用户的推荐结果

输入关键字

提交

薛秋薇

退出



导演: 罗伯特 泽米吉斯 Robert Zemeckis

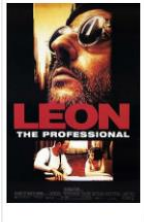
主演: 汤姆 汉克斯 Tom Hanks

年代: 1994

豆瓣评分: 9.5

浏览量: 0 / 收藏量: 0 / 评分量: 14

网站评分: 5.0



### 这个杀手不太冷

怪蜀黍和小萝莉不得不说的故事。.....

导演: 吕克 贝松 Luc Besson


主演: 让 雷诺 Jean Reno

年代: 1994

豆瓣评分: 9.4

浏览量: 0 / 收藏量: 0 / 评分量: 10

网站评分: 1.0



### 美丽人生

最美的谎言。.....

导演: 罗伯托 贝尼尼 Roberto Benigni

主演: 罗伯托 贝尼尼 Roberto Beni...

年代: 1997

图 4-12 基于 als 的推荐

4.6 数据库逻辑结构展示

相关信息如图 4-13 所示:

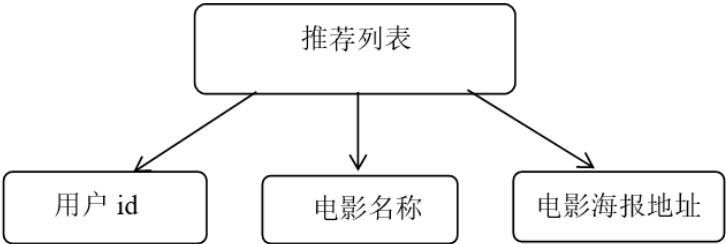


图 4-13 数据库内在逻辑图

#### 4.7 系统 E-R 图展示

本系统的 ER 图如下图 4-14 所示:

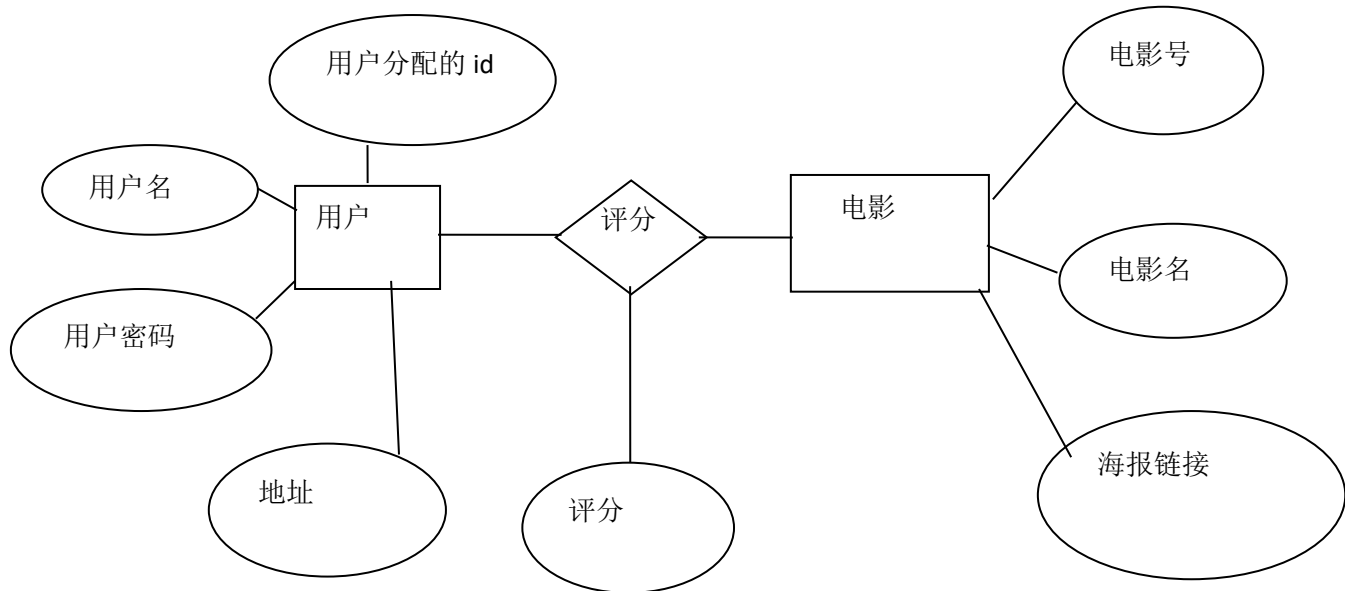


图 4-14 系统 ER 图

## 5 总结与展望

### 5.1 总结

本文的系统包含电影展示、评分模块、用户信息、推荐结果几个相关模块。本系统分析阐释了推荐系统的结构及各功效。介绍并分析了系统各模块，用户需求，以及对数据库，数据表的设计

本系统采用多种改进过的推荐算法：基于用户，基于物品和基于 als 的推荐算法，使得用户的可选择性更高。本文利用 CSS3 语言美化了网页，结合 Django 框架，利用 Django 内置的功能实现了用户注册，与推荐结果的前端展示。使系统结构更为更加整洁。

系统仍然有很多不足，主要有以下的一些内容需要改进：

通过给用户一定的选择权来解决冷启动的问题，比如允许他们自由的选择喜好标签，再结合标签给出特定推荐。再增加一张表将用户的一些行为数据加以记录，统计用户看过得电影，查看用户的评分数据，假使这部电影用户给分比较低，那就在用户的行为数据表中对此电影的分数进行降分处理，分数打的比较高就进行加分处理，再结合标签进行推荐。

## 参考文献

- [1] Pilli L E, Mazzon J A. Information overload, choice deferral, and moderating role of need for cognition: Empirical evidence[J]. Revista De Administracao Publica, 2016, 51(1):36-55.
- [2] 朱杰. 基于标签和als的图片推荐系统[D]. 天津师范大学,2014.
- [3] 王国霞,刘贺平. 个性化推荐系统综述[J]. 计算机工程与应用,2012:66-76.
- [4] 许海玲,吴潇,李晓东,等 互联网推荐系统比较研究[J]. 软件学报. 2009, 20(2): 350-362.
- [5] Hofmann T. Latent semantic models for collaborative filtering[J].ACM Transactions on Information Systems, 2004, 22(1):89-115.
- [6] 杨杰. 个性化推荐系统应用及研究[D].中国科学技术大学,2009.
- [7] Ungar L H, Foster D P. Clustering methods for collaborative filtering[C].Proceedings of the AAAI Workshop on Recommendation Systems,Madison, USA, Jul 26-27, 1998. Menlo Park, CA, USA: AAAI, 1997: 114-129 .
- [8] Tan Xueqing, He Shan. Research Review on Music Personalized Recommendation System[J]. New Technology of Library and Information Service, 2014, 30(9): 22-32.
- [9] Wang X, Rosenblum D, Wang Y. Context-Aware Mobile Music Recommendation for Daily Activities [C]. In: Proceedings of the 20th ACM International Conference on Multimedia. ACM, 2012: 99-108.
- [10]徐晨. 基于二部网络分析的推荐算法研究及其应用[D]. 扬州大学,2017.
- [11]李小浩.als推荐算法稀疏性与可扩展性问题研究[D].重庆大学,2015.
- [12]X. Su, T.M. Khoshgoftaar, A survey of collaborative filtering techniques, Adv. Artif. Intell. 2009 (2009) .
- [13]喜晶. 个性化推荐技术的分析和比较[J]. 软件研发与应用,2016:39.
- [14]Breese J S, Heckerman D, Kadie C. Empirical analysis of predictive algorithms for collaborative filtering[J]. Uncertainty in Artificial Intelligence, 2013, 98(7):43-52.
- [15]项亮.推荐系统实践[M].北京:人民邮电出版社,2012 :26.
- [16]冯阿敏. 基于用户als算法的推荐系统的设计与实现[D]. 西安电子科技大学,2017.
- [17]张明珺. 基于用户的个性化影视推荐系统的研究与实现[D]. 电子科技大学,2017.
- [18]陈诺言. 基于个性化推荐引擎组合的推荐系统的设计与实现[D]. 华南理工大学,2012.

- [19]Pazzani M J,J Muramatsu,D Billsus. Syskill & Webert:Identifying interesting web sites[C].  
Proceedings of the national conference on artificial intelligence,1996:54-61.
- [20]尤方圆. 电影推荐系统的设计与实现[D]. 华中科技大学,2013.

## 致谢

在本次的毕业设计中，非常感谢我亲爱的老师，本次论文从开题开始就一直在老师的悉心教导下写完的，在修改论文以及文献翻译的时候，我的导师非常耐心的一遍一遍的细心检查我的论文以及文献翻译，即使是微末的小细节，老师也一一指出。给了我非常详尽的修改建议，对我帮助非常大，即使是一些细微的格式错误老师也会加以指正，让我及时修改。非常的佩服老师的精益求精的工作态度和严谨的态度，朴实平易近人的人格魅力，我觉得我应该把老师当做学习的榜样。

还有非常感谢在大学四年里朝夕相处的舍友，得以与他们共同经历这大学的宝贵时光我由衷的感到幸福，在一起彼此帮助彼此，万言千语涌上心头，一些有关论文修改的消息也是相互之间共同分享，在这里我也非常的感谢她们热心帮助与支持，是她们给与了我前行的力量。

在这个论文即将完成的时刻，我觉得十分感动，心情久久不能平复，从前期的选题到开题再到论文完成，我希望能尽到我的全部努力与心血。但是在过程中确实遇到了很多困难，曾经使得我非常的困惑。这篇论文的完成离不开我的导师对我的悉心教导，谢谢老师！