本科学生毕业设计

基于协同推荐的视频网站的设计与实现

院系名称： 计算机科学与技术学院

专业班级： 计算机科学与技术系17-4班

学生姓名： 陈盈涛

指导教师： 杨泽雪

职 称： 副教授

企业指导教师：

**黑龙江工程学院**

二○二一年六月

The Graduation Design for Bachelor's Degree

Welding Process

Candidate：Chen Yingtao

Specialty：Computer Science And Technology

Class：17-4

Supervisor：Associate Prof. Yang Zexue

Enterprise guidance：Senior Engineer XY

Heilongjiang Institute of Technology

2021-06·Harbin

摘 要

推荐系统作为大数据盛行的产物，是应对数据过载问题的解决方案，可以帮助解决信息消费用户和信息提供者之间的信息准确、快速匹配问题。推荐系统可以给网站带来直接的流量和经济效益，因此目前世界著名的网站平台公司都在发展推荐系统。

推荐系统搜集用户的日常行为，挖掘信息背后的用户个人偏好，进行物体分类。所谓物以类聚，人以群分，推荐系统要做的内容就是根据规则将物品聚类，将用户分群，然后对每个用户投其所好，满足用户需求。用户行为数据可以是播放记录、购物记录、评分记录、浏览记录、搜索点击次数、评论记录等。推荐系统最核心内容是以何种规则做到物以类聚，人以群分。其中物以类聚对应基于物品的协同过滤，人以群分对应基于物品的协同过滤。

本设计通过应用协同过滤推荐算法，建立一个个性化视频网站，能根据不同兴趣爱好的用户，建立个性化电影推荐清单，满足用户拥有个性化定制主页的需求，设计过程中涉及MovieLens下的协同过滤推荐验证实验，使用Flask架构设计网站过程，利用MongoDB作为持久层数据库，以及利用缓存数据库优化推荐系统效率的过程。

关键词：协同过滤；视频网站；推荐系统；MovieLens；Flask；Nosql

ABSTRACT

As the product of big data, recommendation system is a solution to the problem of data overload, which can help solve the problem of accurate and fast information matching between information consumers and information providers. Recommender system can bring direct traffic and economic benefits to the website, so the world famous website platform companies are developing recommender system.

Video recommendation system collects users' daily behaviors, mines users' personal preferences behind information, and classifies objects. The so-called "birds of a feather flock together, people flock together". The recommendation system clusters the items and the users, and then gives preference to each user to meet their needs. User behavior data can be playback history, scoring information, search hits, comment records, etc. The core content of recommendation system is how to classify, that is, according to what standard, users are divided into groups this is collaborative filtering for users and items are clustered this is collaborative filtering for items.

Through the application of collaborative filtering recommendation algorithm, this design establishes a personalized video website, which can establish a personalized movie recommendation list according to different interests of users, and meet the needs of users with personalized customized home page. The design process involves the collaborative filtering recommendation verification experiment under movielens, and uses the flash architecture to design the website process, Using mongodb as persistence layer database, and using cache database to optimize the efficiency of recommendation system.

**Key words**: Collaborative filtering; Video website; Recommendation system; MovieLens；Flask；Nosql

目 录

[摘 要 1](#_Toc27166)

[ABSTRACT 2](#_Toc6611)

[目 录 3](#_Toc12168)

[第1章 绪 论 1](#_Toc20503)

[1.1 研究背景 1](#_Toc6356)

[1.2 研究目的和意义 2](#_Toc4369)

[1.3 研究现状 3](#_Toc31721)

[1.3.1 推荐算法发展 3](#_Toc8166)

[1.3.2 推荐算法的研究现状 3](#_Toc10446)

[1.3.3 国内外推荐算法应用现状 3](#_Toc2092)

[1.4 本文安排 4](#_Toc12740)

[第2章 相关技术介绍 5](#_Toc31765)

[2.1推荐算法介绍 5](#_Toc11586)

[2.1.1 基于内容的推荐算法 5](#_Toc30988)

[2.1.2 基于内存的协同过滤推荐算法 5](#_Toc2221)

[2.1.3 基于模型的协同过滤推荐算法 5](#_Toc3574)

[2.1.4 混合推荐算法 6](#_Toc31243)

[2.2 相似度算法计算方式 6](#_Toc14515)

[2.2.1 基于距离计算相似度 6](#_Toc20471)

[2.2.2 基于系数计算相似度 7](#_Toc15522)

[2.3 推荐系统离线评测方案 7](#_Toc6827)

[2.3.1 离线实验评测概述 7](#_Toc8744)

[2.3.2 推荐算法评测指标 8](#_Toc15829)

[2.3.3 评分预测评价指标 9](#_Toc20367)

[2.4 本章小结 9](#_Toc14992)

[第3章 推荐算法选择和应用 10](#_Toc28084)

[3.1 基于用户的协同过滤算法介绍 10](#_Toc14783)

[3.1.1 User CF 原理 10](#_Toc8036)

[3.1.2 User CF 实现 10](#_Toc25238)

[3.2 基于项的协同过滤算法介绍 12](#_Toc2129)

[3.2.1 Item CF 原理 12](#_Toc9752)

[3.2.2 Item CF实现 13](#_Toc18699)

[3.3 User CF 和 Item CF 的对比 14](#_Toc12238)

[3.4 本章小结 14](#_Toc32608)

[第4章 协同过滤推荐算法实验与应用 15](#_Toc27917)

[4.1 协同过滤推荐算法实验设计 15](#_Toc5896)

[4.1.1 实验1 基于杰卡德相似度的Item CF 实验 15](#_Toc26494)

[4.1.2 实验2 基于余弦相似度的Item CF实验 17](#_Toc21658)

[4.1.3 实验3 基于杰卡德相似度的User CF 实验 18](#_Toc21696)

[4.2 协同过滤推荐算法实验结果 19](#_Toc5386)

[4.2.1 协同过滤相似性算法分析 19](#_Toc3176)

[4.2.2 推荐算法实验评测 20](#_Toc20041)

[4.3 本章小结 20](#_Toc24230)

[第5章 个性化视频推荐网站的设计 21](#_Toc448)

[5.1 网站技术分析 21](#_Toc22863)

[5.1.1 后端技术介绍 21](#_Toc6072)

[5.1.2 数据库技术 21](#_Toc13869)

[5.1.3 网页前端展示技术 21](#_Toc4846)

[5.2 需求分析设计 22](#_Toc17592)

[5.2.1 视频播放网站前端 22](#_Toc17896)

[5.2.2 用户功能模块 22](#_Toc29347)

[5.2.3 推荐功能 22](#_Toc16320)

[5.2.4 管理员功能 23](#_Toc29166)

[5.3 数据库设计 23](#_Toc5252)

[5.3.1 用户数据表 24](#_Toc12371)

[5.3.2 电影数据表 24](#_Toc1374)

[5.3.3 电影播放记录表 25](#_Toc13128)

[5.3.4 缓存数据表 26](#_Toc28439)

[5.4 本章小结 28](#_Toc17649)

[第6章 个性化视频推荐网站的实现 29](#_Toc22976)

[6.1 开发环境搭建 29](#_Toc430)

[6.2 用户网站功能实现 29](#_Toc6781)

[6.2.1 电影视频播放网站网页设计清单 29](#_Toc4920)

[6.2.2 用户个人信息管理 30](#_Toc16580)

[6.2.3 用户操作行为功能 32](#_Toc3418)

[6.2.4 站内搜索功能 32](#_Toc4760)

[6.2.5 推荐功能 33](#_Toc16849)

[6.3 后台管理功能实现 33](#_Toc14206)

[6.3.1 电影信息管理 33](#_Toc19420)

[6.3.2 电影资源管理 34](#_Toc28740)

[6.3.3 电影播放热度统计 34](#_Toc22903)

[6.3.4 用户信息管理 34](#_Toc27087)

[6.4 网站前端首页功能实现 35](#_Toc10081)

[6.4.1 前端柱形图绘制 35](#_Toc27532)

[6.4.2 前端分页功能实现 35](#_Toc3673)

[6.5 本章小结 35](#_Toc26169)

[结 论 36](#_Toc13108)

[参考文献 37](#_Toc19998)

[致 谢 39](#_Toc14527)

[附 录 40](#_Toc17393)

第1章 绪 论

## 1.1 研究背景

近二十年，随着科技进步，互联网产业的发展，人们踏入大数据时代，互联网已经融入人们的日常生活。人们可以通过智能手机、平板电脑、个人计算机、智能手环及智能手表等网络设备接入到互联网，实现远程管理、远程监控、远程办公、移动支付、网络就医、在线学习、了解时事、信息交流及在线购物等操作，互联网将人类社会从独立的点联结成一张复杂、巨大的网。

由于互联网的网络节点繁多、拓扑结构复杂，因此信息流动复杂且迅速，就信息来源方面，有各大媒体、社交网络、电子商务网站、网络互娱公司、信息传媒平台、网民个人等源头提供大量的信息。在信息消费方面，有网民个体、各大数据分析公司、国家机关、行业公司等都利用大量数据信息满足自身需要。例如亚马逊通过用户个人信息了解用户的个人偏好、收入水平等信息制定公司发展战略，信息数据俨然成为当前大数据时代最重要的战略资源。其次，用户经常会被各种推销消息所打扰，从推荐商品、购物、贷款、买房买车到各种诈骗、虚假信息，究其原因在于个人信息的泄露导致被垃圾数据所困扰。

如何合理、合法的利用数据信息资源，发展经济效益成为时代的主题。从用户个人角度，高效地查找所需信息可以节约个人时间，提高工作、生活效率；从公司发展角度，高效的信息过滤可以带给公司在同行业的竞争优势，提高公司商品的销售额。综上所述,商品和消费者之间如何高效、准确地匹配成为公司的财富密码，推荐系统成为了大数据时代发展方向。

结合电影观影网站的行业，上述问题引申为为电影和观众之间高效、准确地匹配问题，帮助用户找到心仪的电影带来经济效益。假如用户明确自己的需求，可以借助搜索引擎来查找得到自己喜欢的电影进行付费观影，例如用户甲想看《复仇者联盟3》时，可以在网站中搜索电影名进行付费观影；但是如果用户甲需求不明确时，应该如何找到喜欢的电影，即用户想找一部自己喜欢的电影，但是不知道具体是什么，面对网站中数以万计的电影只能漫无目的的搜索，低效的搜索只会浪费用户的耐心导致用户放弃从而带来极差的用户体验，这类大部分用户都会遇到的信息过载（Information Overload）问题。

为了处理信息过载需要有一个根据用户需求能有效快速地为用户推荐信息的系统，它可以将网络信息过滤，将可能有用的信息分发给需要的用户，对于用户而言，可以将大海捞针变成湖里捞针，解决用户的选择困难问题，帮助用户获得良好的体验，从而获得心理上的满足，并且节省用户时间，帮助用户在有限的时间高效的完成目的；对于内容提供者来说，可以更加高效地分发内容，投其所好，赢得用户口碑，获得用户的青睐，节省系统的资源等不必要的消耗，在同行竞争者中占据先机，最终提高经济效益。这样的系统可以帮助内容提供者和内容消费者带来双赢。

## 1.2 研究目的和意义

在日常生活中，可以发现视频网站方总能贴用户心意地推荐用户喜欢的电影、电视节目，这个过程的背后原理是什么，作为内容提供者的视频网站是如何为作为消费者的用户推荐用户喜欢的电影。从信息获取的角度来看，人类获取信息的方法有主动搜索与被动推荐。

以实际经验举例：

主动搜索是：一个用户想要主动了解信息，会先从周围开始询问，再从周围了解到的线索拓展，继续向更大的范围询问，这也是搜索引擎的搜索过程。

被动推荐是：用户周围的朋友主动告诉用户相关信息，并由用户自行筛选，用户作为一个被动接收方不需要额外的搜索时间，为用户推荐信息是推荐系统需要处理的过程，这也是推荐系统的主要原理。

搜索引擎依靠用户手动在庞大的影视资源中搜索自己心仪的电影、电视节目，需要明确的目标，否则会陷入搜索困惑的地步，浪费大量无效时间。推荐系统通过将查找用户喜欢电影这一过程交付给系统进行，为用户节约搜索时间，通俗的说这是一个推荐系统根据条件猜测用户喜好的过程，目前研究旨在提高推荐系统猜测的准确率和效率。

个性化推荐系利用用户数据，使用推荐算法帮助用户查找和筛选电影，为用户提供充足、有效、合适、多样、新颖的个性化电影库。有效的推荐系统可以帮助减少搜索范围，节省搜索时间，提高用户体验。但推荐系统被要求有尽可能快的运行速度，尽可能少的资源占用，既要提高推荐准确率，又要提高推荐速度，这是推荐系统所面对的主要主要矛盾。

本设计通过建设一个小型的视频网站的过程，研究协同过滤推荐算法的原理和实现方法，利用推荐系统建立个性化视频网站，赋予个性化定制的功能，为不同的用户推荐用户所喜欢的电影，同时涉及到研究两种不同相似性算法条件下的推荐算法效果分析、评价过程。

## 1.3 研究现状

### 1.3.1 推荐算法发展

对于推荐算法的研究开始于上世纪，到现在广泛应用于各行各业。从电商、视频、直播、新闻、媒体到广告、娱乐、游戏等都有推荐系统的影子，通过大量分析用户行为数据，给用户提供个性化服务来提高网站流量、吸引更多的用户、提高市场占有率，最终提高经济效益，个性化推荐系统应用于购物类网站、音乐类网站、电影和视频类网站、阅读类网站、社交和新闻类网站、小视频和直播类网站等，随着行业发展需求不断的变化，对推荐系统的要求越来越高，对推荐系统的研究越来越严格。推荐系统按照角色组成可以分为：

1、用户角色，即推荐的目标人群；

2、内容角色，即待推荐的主要内容；

3、推荐角色，将内容推荐给用户的规则，是连接用户与内容的核心部分，也是需要深入研究的主要问题。

### 1.3.2 推荐算法的研究现状

协同过滤是一种应用较广的成熟推荐算法，目前在工业生产中被广泛采用。其基本思想是根据用户历史行为数据推荐用户喜好，具体有三个流程：

1、数据处理，对于用户历史行为数据的数据处理。

2、相似性处理，根据历史数据进行相似性计算，确定相似性关系。

3、推荐，根据相似性规则关系为用户进行定制。

目前在推荐系统的发展中长期存在着某些问题需要去解决：

1、冷启动，在没有历史数据的情况下如何进行推荐。

2、启动数据稀疏，导致构建的相似性关系矩阵稀疏，从而影响推荐结果普遍偏低，不易于比较。

3、用户兴趣漂移是指用户的兴趣变化，因为用户是具有思维的个体，具有不确定性和偶然性，用户的喜好会发生突变，此时原先的相似性矩阵就无法适用。

### 1.3.3 国内外推荐算法应用现状

个性化推荐系统在电影和视频网站中具有重要的应用地位，因为电影视频资源的数量庞大、不容易遍历搜索，导致用户对于海量信息的搜索存在巨大的困难。

腾讯视频网站的推荐系统，可以根据电影相似性关系规则进行物品聚类，帮用户筛选可能感兴趣的电影；也可以根据用户好友的之间的联系，利用QQ、微信中的好友数据资源为用户分群，以群来进行推荐，在用户好友圈子中互相分享受欢迎的电影节目。QQ音乐平台的推荐，主要依据用户的历史记录和好友圈的历史记录来分析，通过比对所有人音乐的喜好，计算每首音乐之间的相似性，结合用户的听歌记录生成用户的私人歌单，同时也可以通过找到用户有共同爱好的好友，互相推荐音乐。阅文网站的小说推荐，当用户浏览一本小说时，系统会给出当前小说相似的小说列表，从相似性出发找到同类型的小说加入到推荐列表推荐给用户。

电商应用方面以亚马逊举例，推荐系统在电商的发展主要得益于亚马逊，通过在数以亿计的购买记录数据中提取用户喜好信息，用数据来刻画出来用户的喜好，对相似同类人、商品进行归类，掌握每一个人喜好，利用用户喜好生成推荐列表，提高用户购买率，给公司带来直接效益。

## 1.4 本文安排

第1章，绪论，简要说明本设计研究背景、原因意义、目的以及现状

第2章，推荐算法、相似度算法的基础介绍和评价方案了解。

第3章，介绍两种协同过滤推荐算法优势，举例说明实现过程。

第4章，设计两种协同过滤推荐算法实验，确定最后采用的方法。

第5章，进行网站网站的需求分析、功能设计及数据库设计规划。

第6章，个性化视频网站功能实现。

1. 相关技术介绍

当前主流的推荐系统是采用多种算法混合的混合推荐系统，作为一种综合性推荐系统，是融汇各种推荐算法优点来设计，例如：基于内容的推荐系统（Content Based, CB）、使用历史数据的协同过滤推荐系统（Collaborative Filtering,CF）使用机器学习的基于模型的协同过滤算法（Model-based Collaborative Filtering）等。因为每种推荐算法都有其应用范围、限制，在不同的条件下，推荐效果各有差异，因此开发混合推荐系统成为当前的主流的研究方向。

## 2.1推荐算法介绍

### 2.1.1 基于内容的推荐算法

该算法主要用于文本类项目中，采用浅层模型分析，提取核心内容词汇，根据物品的便签属性、内容关键词等，与用户兴趣描述标签配对，推荐有相似标签的物品给用户。例如：某用户喜欢看《夏洛特烦恼》，可以给用户打上喜欢看喜剧、喜欢沈腾的电影作为标签，根据这信息可以为该用户推荐同为喜剧的《人在囧途》、《飞驰人生》等，这三部电影的内容标签为：影片类型喜剧，主演沈腾、尹正。再者假设一用户在网上购买了《数据结构》书籍，根据这一信息，为改用户打上计算机学习者、研究数据结构的标签，可以推荐给该用户《算法》书籍，这两本书籍同为：同为计算机指导用书，并且算法和数据结构知识具有重叠性。基于内容的推荐是根据物品本质进行推荐，推荐过程比较直观，推荐理由充分，需要做的工作是物品的归类、内容标签的标定，于此同时存在推荐精度较差，惊喜度不足的问题。

### 2.1.2 基于内存的协同过滤推荐算法

作为当前推荐算法中比较常用和经典的主流算法之一，目前在商业项目中具有较多应用，基本原理就是根据用户的历史行为数据归类出相似性，发现用户、物品间的相关性进行推荐，如果用户喜欢A，那么就推荐和A相似的物品B推荐给用户。基于内存的协同推荐算法的计算过程全放在内存之中 ，虽然实现简单、直观，但是不能为用户实时推荐，每次有新内容加入时需要重新计算，难以应对高并发的需求。

### 2.1.3 基于模型的协同过滤推荐算法

该算法通过机器学习的方法预测用户对商品的喜好程度，主要的步骤有线下的模型训练、线上的推荐预测，利用数据集训练出较好的推荐算法模型，预测新样本时可以通过从新内容的特征参数输入到推荐算法模型中得到预测结果。常规的应用方法有关联分析、回归算法、聚类算法、分类算法、隐语义模型、神经网络、图模型、矩阵分解等。有着适用范围广大，推荐灵活，可以应对数据稀疏的问题，能很好地发现用户潜在偏好。

### 2.1.4 混合推荐算法

在上文都是单一的推荐算法，每个算法都有各自的有点和缺点，在生产中为了应对复杂环境需要取长补短，因此产生混合推荐算法，混合推荐算法不是一种具体的算法，而是多种推荐算法结合得到的新的推荐算法，虽然提高了算法的复杂度，但是可以利用各种算法的互补提高推荐系统健壮性。

## 2.2 相似度算法计算方式

推荐算法中具有一个核心的问题：相似度的判断，如何衡量两个项之间的相似性，需要将抽象的关系转化为一个具象的数据，通过建立项的数学模型转化为向量，通过向量之间的相似性来比较，常见的相似性算法如下文所示。

### 2.2.1 基于距离计算相似度

1、欧几里得距离公式，由平面直角坐标系两点间距离公式推广产生，其含义为n维空间坐标系中两个点的绝对距离，通过构建n维向量可以简单地计算距离，是一种非常方便的计算方式，具体计算过程如公式（1）所示。

公式（1）

但其缺点是容易受到尺度影响，即计算的距离会因为特征维度的单位，导致结果异常。例如在二维空间坐标系中，x为身高，单位是厘米（cm），y为体重，单位是千克（kg），这两个单位不同的指标计算的结果会发生偏移，导致结果没有参考价值。因此在使用欧式距离度量之前，需要对数据进行归一化处理，在单位尺度不同的条件下使用距离公式都需要特别注意尺度问题。

1. 曼哈顿距离，别称出租车距离、国际象棋距离，是简化欧式距离衍生出的公式，由于欧式距离中存在大量平方、根号操作导致计算时间长，因此将欧式距离利用绝对值进行简化，具体计算过程如公式（2）所示。

公式（2）

3、切比雪夫距离，在空间坐标系中两点之间所有对应坐标差值取绝对值后的最大值，具体计算过程如公式（3）所示。

公式（3）

4、闵可夫斯基距离，一种距离的定义，其计算过程如公式（4）所示，是对多种距离度量公式的概括性表述。

公式（4）

### 2.2.2 基于系数计算相似度

1、余弦相似度，通过计算两个向量的夹角余弦值来评估他们的相似系数，具体计算过程如公式（5）所示，计算求两个向量之间夹角余弦值，作为相似性，余弦值与相似度成正相关。

公式（5）

1. 杰卡德系数，利用两个集合A、B的交集元素数量与并集元素数量之比确定集合的相似度，具体计算过程如公式（6）所示。

公式（6）

## 2.3 推荐系统离线评测方案

### 2.3.1 离线实验评测概述

离线实验通常用于开发阶段的测试，使用离线数据集评测推荐算法的推荐效果，具体过程如下：

1、获得用户行为数据。

2、将数据集按照一定的规律分成训练集和测试集。

3、利用训练集训练推荐模型。

4、利用离线评测指标在测试集上进行预测结果的评价。

5、根据评测结果对于推荐算法进行改进。

离线实验的优点是：

1、不需要用户参与、时间经济成本低。

2、利用离线数据验证、时间周期短、效率高。

3、可以无人值守，调试修改算法灵活。

离线实验的缺点是：

1、受到数据集合影响大，例如数据集稀疏、片面等局限性。

2、对于评价结果缺乏主观性，其原因在于用户思维不易具体数值化。。

3、对于用户的变化无法做到及时响应，例如用户的兴趣偏移。

### 2.3.2 推荐算法评测指标

推荐系统本质上属于一种分类模型，因此可以采用分类模型评价的指标对推荐系统的结果进行评测，而混淆矩阵是评判分类结果优劣的方案之一。

1、混淆矩阵及其相关指标。在分类实验中将真实结果、预测结果的真假共四种结果分布到2 x 2的表格中进行分析，如表2-1所示。

表2-1 混淆矩阵

|  |  |  |
| --- | --- | --- |
| 真实结果 | 预测结果 | |
|  | 真 | 假 |
| 真 | True Positive | True Negative |
| 假 | False Positive | False Negative |

混淆矩阵基础指标有：

TP，将真值预测为真，真正类，是正确预测。

FN，将假值预测为假，真假类，是正确预测。

TN，将真值预测为假，伪假类，是错误预测。

FP，将假值预测为真，伪正类，是错误预测。

衍生出二级指标对预测结果评价：

1. 准确率：预测正确的数量（真正类+真假类）与所有预测数量之比，具体计算过程如公式（1）所示。

公式（1）

1. 精确率：将真类预测正确的数量（真正类）与所有预测为正确的数量（真正类、伪正类）之比，具体计算过程如公式（2）所示。

公式（2）

1. 召回率：将真类预测正确的数量（真正类）与所有真类的数量（真正类、伪假类）之比，具体计算过程如公式（3）所示。

公式（3）

2、ROC曲线和AUC面积

ROC（Receiver Operating Characteristic）曲线和AUC（Area Under the Curve）面积常常用来评价一个二值分类器的优劣。ROC曲线是以FPR（假正例率）做横坐标、TPR（真正例率）做纵坐标绘制出的曲线。假正例率伪正类的数量在所有预测为假类的数量的比例，真正例率是真正类的数量在所有预测为正类的数量的比例，具体计算过程如公式（4）、（5）所示，AUC面积是ROC曲线向下、向右与x、y轴围成的面积。

公式（4）

公式（5）

3、F分数，是精确率和召回率在不同权重下的妥协结果，是为了解决精确率和召回率不能同时满足的问题做出的取舍，当精确率重要时，提高精确率权重，反之提高召回率权重，具体计算过程如公式（6）所示。

公式（6）

F1 score，此时β=1，精确率和召回率权重相同，两者一样重要，由公式（6）与β=1推导出公式（7）。

公式（7）

### 2.3.3 评分预测评价指标

对于推荐结果最终会收到一个用户的评分，可以利用用户对推荐结果的评分和推荐系统预测的评分进行一个误差计算来评估推荐系统效果的优劣。具体误差评价方法有平均绝对误差（MAE），均方根误差（RMSE）等。其系数结果越小，说明系统推荐准确性更高。具体计算过程如公式（8）、公式（9）所示。

公式（8）

公式（9）

## 2.4 本章小结

本章主要介绍推荐算法的理论入门基础知识，常见的算法类型、原理，以及算法涉及到的理论数学基础知识和核心相似性算法数学原理及推荐算法最后的统计评价标准，推荐算法需要综合性的数学学科基础。

第3章 推荐算法选择和应用

基于内存的协同过滤（Memory-Based Collaborative Filtering）可以细分为基于用户的协同过滤推荐（User-based Collaborative Filtering）、基于项的协同过滤推荐（Item-based Collaborative Filtering）。

## 3.1 基于用户的协同过滤算法介绍

### 3.1.1 User CF 原理

User CF基本思想是根据用户历史行为记录发现兴趣偏好相似的邻居用户群，根据邻居用户群体的行为记录的并集来为用户推荐，如图3.1，甲和乙观影记录相似都看过电影1、电影5，甲的观影记录为（电影1、电影3、电影5），乙的观影记录为（电影1、电影4，电影5），可以得出甲、乙的偏好相似，甲、乙这类用户喜好观看电影1、电影3、电影4、电影5，因此在甲、乙之间互相推荐得效率更高，故给甲推荐电影4，给乙推荐电影3。



图3.1 User CF 用户喜好电影图

User CF接近于用户兴趣群体推荐，先发现用户兴趣圈，再在圈内交流兴趣，比较精准，适用物品多且杂的场景，能跨越物品类别推荐的能力，能发现用户潜意识里尚未察觉的兴趣，惊喜度高，能够推荐抽象产品的跨领域能力。

### 3.1.2 User CF 实现

1、根据用户的历史行为，收集用户-电影评分数据记录表如表3-1所示，在该表基础上建立用户-电影评分矩阵如表3-2所示，（数据仅举例用）。

表3-1 用户-电影评分数据记录表

|  |  |  |
| --- | --- | --- |
| 用户编号 | 电影编号 | 评分 |
| 1 | 1 | 4 |
| 1 | 3 | 4 |
| 1 | 6 | 4 |
| 2 | 3 | 5 |
| 2 | 5 | 5 |
| 2 | 7 | 3 |
| 2 | 10 | 5 |
| 3 | 1 | 4 |
| 3 | 5 | 5 |
| 3 | 3 | 5 |
| 4 | 3 | 5 |
| 4 | 2 | 5 |
| 4 | 5 | 3 |

表3-2用户-电影评分矩阵

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 用户编号 | 该用户看过电影的评分 | | | | | | |
|  | 1 | 3 | 6 | 5 | 7 | 10 | 2 |
| 1 | 4 | 4 | 4 |  |  |  |  |
| 2 |  | 5 |  | 5 | 3 | 5 |  |
| 3 | 4 | 5 |  | 5 |  |  |  |
| 4 |  | 5 |  | 3 |  |  | 5 |

2、利用表3-2用户-电影评分矩阵结合余弦系数计算方法，确定用户间相似性矩阵如表3-3所示。

表3-3用户间相似性矩阵

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用户编号 | 1 | 2 | 3 | 4 |
| 1 | 1 | 0.33502969713 | 0.64333315709 | 0.39589732744 |
| 2 | 0.33502969713 | 1 | 0.68931234948 | 0.59083915670 |
| 3 | 0.64333315709 | 0.68931234948 | 1 | 0.69333333333 |
| 4 | 0.39589732744 | 0.59083915670 | 0.69333333333 | 1 |

3、根据用户的相似度度矩阵，可以得出目标用户的相似邻居，由邻居用户的行为记录中筛选出推荐电影给目标用户。

## 3.2 基于项的协同过滤算法介绍

### 3.2.1 Item CF 原理

Item CF 主要考虑项与项之间的联系，通过确定相似的项目集合，然后根据用户历史记录寻找相似的项进行推荐，如图3-2所示，电影1、电影5都有相同的关注甲、丙，所以可得出电影1、电影5具有一定相似，用户乙看过电影5，所以推荐给他和电影5相似的电影1。



图3.2 用户观影记录图

Item CF则更接近个性化推荐，适用于物品较少的场合（便于建立物品相似矩阵），应用场景有豆瓣的豆瓣猜、淘宝的猜你喜欢、百度的猜你想搜、各大音乐播放器的猜你想听及各大视频网站的猜你想看等，同时Item CF还可以给推荐提可信的推荐理由。

例如：淘宝上最近购买物品A的人也购买了物品B，亚马逊的收藏物品A的人在搜索物品B。因为同时购买物品A、B的行为比较多，因此判定A、B间具有很大的相似性，推荐理由充分。

基于项目的推荐算法尤其在电商行业应用广泛，通过分析所有用户对物品的偏好找到物品之间的相似性联系，构建物品相似性矩阵，借助用户的物品行为记录信息来为用户推荐相似物品，推荐精度高，倾向于推荐物品比较稳定的场景，能较快得出在线结果，但是对于物品量多、更新快的场景效果不足，并且在物品数据稀疏，难以构建有效的物品相似度矩阵时，效果较差。

### 3.2.2 Item CF实现

1、首先导入用户-电影评分数据记录表如表3-4所示，并通过该表生成 电影-用户评分矩阵如表3-5所示。

表3-4 用户-电影评分数据记录表

|  |  |  |
| --- | --- | --- |
| 用户编号 | 电影编号 | 评分 |
| 1 | 1 | 4 |
| 1 | 3 | 4 |
| 1 | 6 | 4 |
| 2 | 3 | 5 |
| 2 | 5 | 5 |
| 2 | 7 | 3 |
| 2 | 10 | 5 |
| 3 | 1 | 4 |
| 3 | 5 | 5 |
| 3 | 3 | 5 |
| 4 | 3 | 5 |
| 4 | 2 | 5 |
| 4 | 5 | 3 |

表3-5 电影-用户评分矩阵

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 电影编号 | 看过该电影的用户评分 | | | |
|  | 1 | 2 | 3 | 4 |
| 1 | 4 |  | 4 |  |
| 2 |  |  |  | 5 |
| 3 | 4 | 5 | 5 | 5 |
| 5 |  | 5 | 5 | 3 |
| 6 | 4 |  |  |  |
| 7 |  | 3 |  |  |
| 10 |  | 5 |  |  |

1. 根据上述电影-用户评分矩阵中的数据，结合余弦相似性计算方式，得到电影与电影之间的相似矩阵，电影相似性矩阵如表3-6 所示。

表3-6 电影相似性矩阵

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 电影编号 | 1 | 2 | 3 | 5 | 6 | 7 | 10 |
| 1 | 1 | 0 | 0.6671 | 0.4603 | 0.7071 | 0 | 0 |
| 2 | 0 | 1 | 0.5241 | 0.3906 | 0 | 0 | 0 |
| 3 | 0.6671 | 0.5241 | 1 | 0.8871 | 0.4193 | 0.5241 | 0.5241 |
| 5 | 0.4603 | 0.3906 | 0.8871 | 1 | 0 | 0.6509 | 0.6509 |
| 6 | 0.7071 | 0 | 0.4193 | 0 | 1 | 0 | 0 |
| 7 | 0 | 0 | 0.5241 | 0.6509 | 0 | 1 | 1 |
| 10 | 0 | 0 | 0.5241 | 0.6509 | 0 | 1 | 1 |

3、根据电影相似性矩阵，结合用户个人观影记录，生成推荐电影列表。

## 3.3 User CF 和 Item CF 的对比

通过User CF和Item CF的流程对比可知，User CF、Item CF实现过程大体相似，其中得到的核心相似性矩阵不同，具体异同如表3-7所示。

1、从计算主体上来看，User CF计算用户之间的相似性，最终得到用户与用户之间的相似性矩阵，其性能与用户数量有关。而Item CF计算电影之间的相似性，最终得到电影与电影之间的相似性矩阵，计算量取决于电影数量。

2、从应用领域上来看，User CF倾向于找到用户临近的聚类，即找到共同爱好用户群体，常与社交功能结合。Item CF则是为电影寻找聚类，对于电影的相似较为敏感，故常用于电商等基于物品推荐的领域。

3、从实时性上看，User CF是根据用户的近邻用户来推荐，对于用户行为发生变化的场景，推荐结果变化不够迅速，适合离线推荐。Item CF是根据用户历史行为进行推荐，可以增加用户新记录的权重，加快对用户新行为记录的响应，当用户行为发生变化时，推荐结果可以快速变化。

4、ItemCF推荐理由较为充分，而UserCF则较为薄弱。

## 3.4 本章小结

本章介绍基于用户的协同过滤算法和基于项目的协同过滤算法的实现过程，以及两者之间的差异不同之处，加深对两种算法的理解，对于协同过滤推荐算法来说，两种实现方式都是具有实际价值的，但是由于两种方法的不同特性，使其具有不同场景下的不同效果，经过流程对比发现，两者区别在于生成相似性矩阵的不同，基于用户的协同过滤需要寻找用户之间的相似，而基于项的协同过滤需要寻找相似物品，两者相似性矩阵不同，生成相似性算法也不同。

第4章 协同过滤推荐算法实验与应用

## 4.1 协同过滤推荐算法实验设计

本设计实验采用环境如表4-1所示。

表4-1 实验环境配置表

|  |  |
| --- | --- |
|  | 环境配置 |
| 操作系统 | Windows 10 家庭版64位 |
| 处理器 | Intel (R)Core(TM) i5-7300HQ |
| 硬盘 | 300 G固态 |
| 内存 | 16 G |
| 编程语言 | Python 3.7 |
| 数据集 | MovieLens 100k数据集 |

MovieLens数据集是由GroupLens实验室发布的电影评分数据，其中包括610多个用户9050多部电影，约有10万条用户评分记录信息，实验过程中使用随机方法划分训练、测试数据。

Python作为当前热门的编程语言，代码语义清晰简短、方便编写人员理解配置，可以移植性强、能跨平台使用，具有强大的数据处理能力，有着非常方便的技术支持，例如Numpy开源科学计算库、pandas数据分析库等。

### 4.1.1 实验1 基于杰卡德相似度的Item CF 实验

实验过程如下：

1、从用户历史评分数据表读入用户的电影评分记录，统计为如表4-2所示的用户-电影矩阵，该表目的是快速发现每个用户的看过的电影以及该用户对该电影的评分。

表4-2 用户-电影矩阵

|  |  |
| --- | --- |
| 用户编号（字典key值） | 该用户看过的电影（字典value） |
| user1 | movie1, movie2 ······ |
| user2 | movie1, movie3, movie4 |
| user3 | movie1, movie3, movie8 |
| user4 | movie2, movie4, movie9 |

2、遍历历史数据，建立如表4-3所示的电影-用户矩阵，梳理每部电影的观影者和评分数据，为寻找电影的观影者交集、并集提供快速途径。

表4-3 电影-用户矩阵

|  |  |
| --- | --- |
| 电影编号（字典key值） | 看过该电影的用户集合（字典value） |
| movie1 | user1,user2,user3,user4 |
| movie2 | user2,use3,user4,user5,user6 |
| movie3 | user5,user6,user9 |
| movie4 | user3,user6,user7 |

3、利用电影之间用户集合数据的交集数目比并集数目（即杰卡德算法）算出电影之间的相似性，核心计算过程如代码4-1所示，生成相似性矩阵如表4-4所示。

代码 4-1 杰卡德系数计算代码

|  |
| --- |
| def jacard(u1,u2):  return len(set.intersection(u1,u2))/len(set.union(u1,u2)) |

表4-4 电影相似性矩阵

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 电影编号 | movie1 | movie1 | movie1 | movie1 |
| movie1 |  |  |  |  |
| movie2 |  |  |  |  |
| movie3 |  |  |  |  |
| movie4 |  |  |  |  |

4、根据表4-4 电影相似性矩阵，结合表4-2 用户-电影矩阵，找到与用户已观看电影相似的电影生成推荐电影列表如表4-5所示，核心程序如代码 4-2。

代码 4-2 基于项的推荐代码

|  |
| --- |
| def recommend(user, K, N, trainSet, movie\_sim\_movie):  rank,watched\_movies = {}, trainSet[user]  for movie, rating in watched\_movies.items():  for related\_movie, w in sorted(movie\_sim\_movie[movie].items(), key=itemgetter(1), reverse=True)[:K]:  if related\_movie in watched\_movies:  continue  rank.setdefault(related\_movie, 0)  rank[related\_movie] = w \* float(rating)  return sorted(rank.items(), key=itemgetter(1), reverse=True)[:N] |

表4-5 推荐电影列表

|  |  |
| --- | --- |
| 电影编号（字典key值） | 预测电影评分 |
| movie1 |  |
| movie2 |  |
| movie3 |  |
| movie4 |  |

由于杰卡德相似度无法体现用户评分数据，因此利用电影相似度和已观看相似电影评分估计出推荐电影评分，最终按预测评分从高到低排序给出推荐结果。

### 4.1.2 实验2 基于余弦相似度的Item CF实验

实验过程如下：

1、从用户历史评分数据表读入用户的电影评分记录，统计为如表4-2所示的用户-电影矩阵，该过程与实验1相同。

2、建立如表4-6 所示的电影-用户矩阵时，与实验1过程相似，区别在于需要对缺省值补0，需要保证每部电影的用户集合满足n维向量。

表4-6电影-用户矩阵

|  |  |
| --- | --- |
| 电影编号（字典key值） | 看过该电影的用户集合（字典value） |
| movie1 | user1,user2,user3,user4,0,0,0,0,0 |
| movie2 | 0,user2,use3,user4,user5,user6,0,0,0 |
| movie3 | 0,0,0,0,user5,user6,0,0,user9 |
| movie4 | 0,0,user3,,0,0,user6,user7,0,0 |

3、利用余弦相似度公式计算电影之间相似度，核心程序如代码4-3所示，生成相似性矩阵如表4-4 所示，除计算代码外，其余与实验1相同。

代码 4-3 余弦系数计算代码

|  |
| --- |
| def cosine(movie1,movie2):  x = to\_array(movie1)  y = to\_array(movie2)  xy = np.sum(x\*y)  x = np.linalg.norm(x,2)  y = np.linalg.norm(y,2)  return xy / (x\*y) |

4、根据表4-4 电影相似性矩阵，结合表4-2 用户-电影矩阵，找到与用户已观看电影相似的电影生成推荐电影列表如表4-5所示，核心程序如代码 4-2。

### 4.1.3 实验3 基于杰卡德相似度的User CF 实验

实验过程如下：

1. 从用户历史评分数据表读入用户的电影评分记录，统计为如表4-7所示的用户-电影表，与实验1、2相同，复述仅为保证流程完整性。

表4-7 用户-电影矩阵

|  |  |
| --- | --- |
| 用户编号（字典key值） | 该用户看过的电影 |
| user1 | movie1, movie2 ······ |
| user2 | movie1, movie3, movie4 |
| user3 | movie1, movie3, movie8 |
| user4 | movie2, movie4, movie9 |

2、根据用户之间交集与并集的比值计算用户间相似度，如代码4-1所示，最终生成用户间相似度，如表4-8所示。

表4-8 电影相似性矩阵

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 电影编号 | user1 | user1 | user1 | user1 |
| user1 |  |  |  |  |
| user2 |  |  |  |  |
| user3 |  |  |  |  |
| user4 |  |  |  |  |

3、根据表4-8 电影相似性矩阵结合表4-7 用户-电影矩阵生成推荐电影列表，生成电影的推荐过程如代码4-4所示，生成的推荐列表如表4-9所示，最终按照预测评分从高到低生成推荐清单。

代码 4-4 基于用户推荐过程代码

|  |
| --- |
| def recommend(user, K, N, trainSet, user\_sim\_matrix):  rank = {}  watched\_movies = trainSet[user]  for v, wuv in sorted(user\_sim\_matrix[user].items(),key=itemgetter(1), reverse=True)[0:K]:  for movie,rating in trainSet[v].items():  if movie in watched\_movies:  continue  rank.setdefault(movie, 0)  rank[movie] = wuv \* rating  return sorted(rank.items(), key=itemgetter(1), reverse=True)[0:N] |

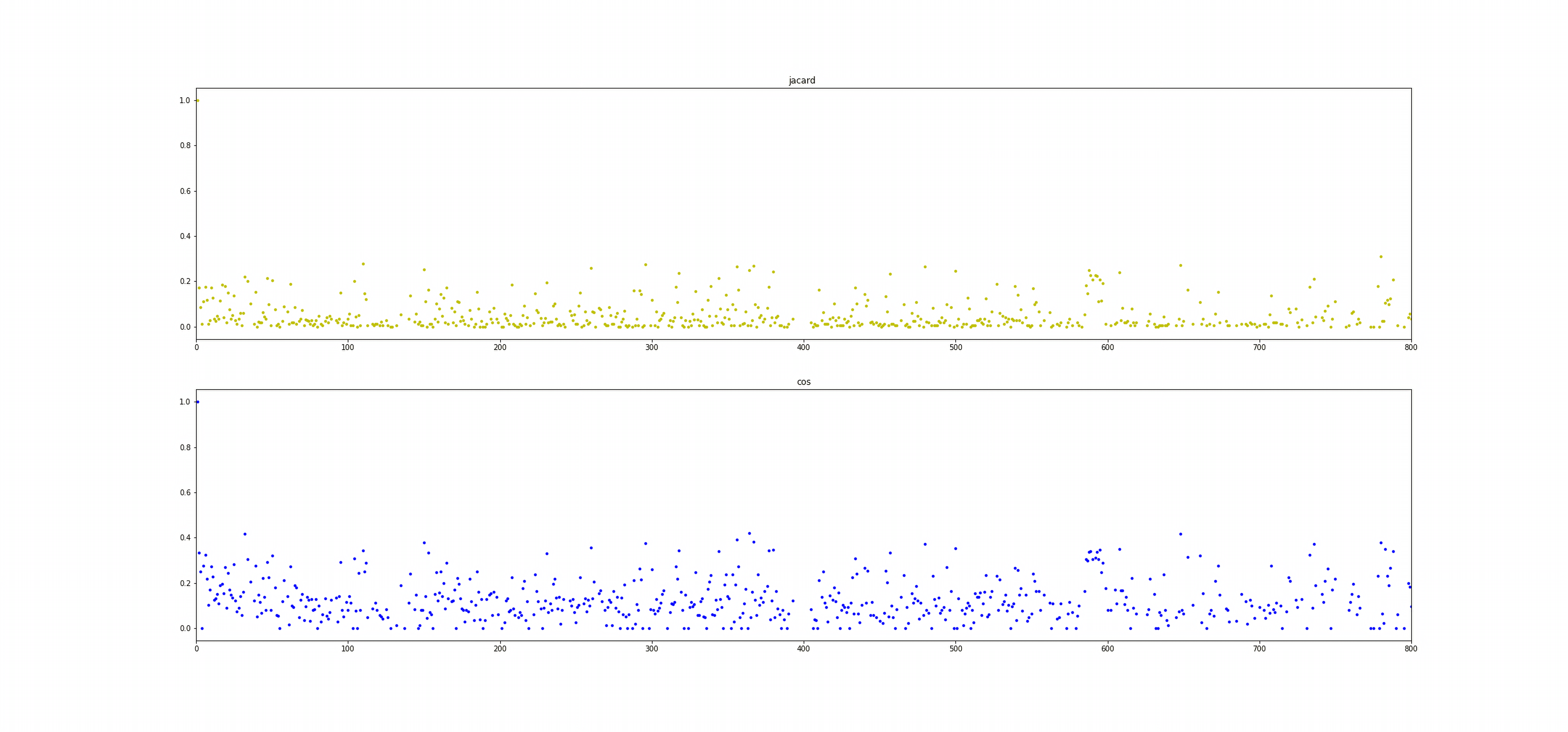
表4-9 推荐电影列表

|  |  |
| --- | --- |
| 电影编号（字典key值） | 预测电影评分 |
| movie1 |  |
| movie2 |  |
| movie3 |  |
| movie4 |  |

## 4.2 协同过滤推荐算法实验结果

### 4.2.1 协同过滤相似性算法分析

1、相似性算法结果，图4.1是编号为1的电影与其他800部电影的相似度值利用matplotlib工具绘制成的散点图，图中上方为杰卡德相似度计算的结果，下方为余弦相似度计算的结果。

图4.1 余弦相似度和杰卡德系数的相似度值对比

由图4.1可以观察：余弦系数与杰卡德系数计算出的相似度值变化相近，波动的峰谷趋势相似，由此可得两者计算结果具有同步性。不同之处在余弦系数计算的结果中的相似度结果结果散布在0到0.4之间，散布区间大，区分度比较高，而杰卡德系数的结果散布0到0.2之间，结果间区分度稍低。

2、对于三个实验方案的运行时间分析：（注：实验1基于项的杰卡德系数，实验2基于项的余弦相似度，实验3基于用户的杰卡德系数）

表4-10 实验用时统计

|  |  |  |  |
| --- | --- | --- | --- |
|  | 实验1 | 实验2 | 实验3 |
| 开始时间 | 21:12:40.193205 | 12:47:20.921754 | 08:24:47.405325 |
| 结束时间 | 21:15:00.385559 | 13:14:54.359994 | 08:24:51.146901 |
| 用时 | 00:02:20.192354 | 00:27:33.438240 | 00:00:03.741576 |

从代码中分析，杰卡德相似性主要依靠计数，余弦相似性则需要进行余弦值的计算，在9000级别的电影数据中，余弦相似度的用户久，在杰卡德和余弦相似结合的实验中，实验时间长，切相似度分布没有明显提升。在基于用户的实验中，用户数量级为600左右，运行效率快，由此可看，在物品多余用户的情况下使用基于用户的协同过滤推荐效率较高。

### 4.2.2 推荐算法实验评测

表4-11 推荐结果评测

|  |  |  |  |
| --- | --- | --- | --- |
|  | 实验1 | 实验2 | 实验3 |
| MAE | 1.608143591 | 1.360856857 | 0.617993205 |
| RMSE | 2.778067166 | 1.851931385 | 0.381915601 |
| 精确率 | 0.1 | 0.05 | 0.05 |
| 召回率 | 0.039215686 | 0.01818 | 0.01785 |

从表4-11结果中分析，结果的精确率低，通过多次实验和算法过程的验证，猜测精确率低的原因在于训练数据不足，相似性结果普遍偏低，影响推荐结果精度。召回率较低原因在于相似性精度不足和只推荐有限部电影，对于召回率有一定影响。

## 4.3 本章小结

本章验证了协同过滤推荐算法基于项和用户的实验，以及两种相似性算法的相似性矩阵结果，由实验综合可知，两种相似性计算方法得到的结果近似相同，但是时间效率属杰卡德算法领先，因此在实验中确定使用基于杰卡德的协同过滤算法作为网站的支撑算法。在该算法下进行项目的推荐系统搭建和推荐系统的调试优化。

第5章 个性化视频推荐网站的设计

## 5.1 网站技术分析

### 5.1.1 后端技术介绍

Flask是目前较流行的Web微框架，基于Python实现的轻量级框架，核心简单、方便入手、易于扩展。默认的Flask不包括诸如数据库支持、表单验证等常见功能，但支持扩展方式添加额外功能，开发者可以根据自己的需求，选择自己的功能模块，因此Flask在Web开发方面变得非常流行。

本设计中推荐算法采用Python实现，因此采用Flask作为开发视频网站的技术支持，易于结合。

### 5.1.2 数据库技术

MongoDB是C++语言开发、基于分布式文件存储的高性能键值对文档型数据库，易部署，存储效率高，存储模式自由，可以存储不同结构的文件，对于不规则的数据存储有较好的支持，对于项目中的数据文件以BSON格式存储（Json格式的拓展）和Python有较好的适配，减少数据转换的代价和风险，适合作为本设计中网站的后台数据库。

Redis是一个将数据以键值对存储在内存中的开源的高性能内存数据库，基于内存和哈希算法，读写速度快，被广泛应用在缓存功能中，亦可作为分布式锁使用，本设计中利用Redis作为缓存数据库。

本设计单独搭建虚拟机Linux服务器，使用CentOS7部署数据库服务器，利用shell脚本实现数据库的基础部署管理操作，提高数据库搭建过程的效率，利用python脚本进行数据库数据的初始数据填入，提高测试效率。如何需要模拟局域网下的生产环境，可以搭建数据库服务器、本地程序服务器、虚拟DNS服务器实现生产环境模拟。

### 5.1.3 网页前端展示技术

视图展示功能使用JinJa2模板和Bootstrap作为网页技术支持。开发响应式布局使用的Bootstrap，由Twitter的工程师为提高内部效率而开发，是一个易用、灵活、可扩展的前端工具集。Jinja2是为Flask开发的基于Python的前端页面模板引擎，设计思想采用Django的模板引擎，轻量化与Flask的微框架搭配使用作为前端视图展现层。

## 5.2 需求分析设计



图5.1网站功能模块图

### 5.2.1 视频播放网站前端

登录相关页面：用户登录、注册等功能，需要增加用户数据前端验证。

分页展示页面：需要前端具有数据分割、分页展示、刷新的功能。

播放评分页面：需要进行电影播放、评分的功能。

数据展示页面：

1、用户登录主页的数据展示，需要根据用户来展示不同的推荐电影。

2、用户搜索电影的结果网站展示。

3、用户管理播放评分数据的展示。

后台管理页面：

1、具有信息展示、搜索功能。

2、具有数据可视化的功能。

### 5.2.2 用户功能模块

用户登录与注册功能，验证用户数据合法性，增加用户信息保密性，允许用户保持登录状态，使用安全较高的Token进行登录状态维护。

用户修改个人信息、登录密码等，确保用户信息安全验证。

播放记录管理，允许用户删除个人的播放评分记录，但不允许修改。

用户播放评分功能，观看影片、对电影打分，需要处理视频数据存储问题，在服务端简要设立文件下载功能。

用户搜索，可以利用条件进行搜索。

### 5.2.3 推荐功能

用户喜好电影推荐：用户进入网站首页时，推荐系统根据登录的用户，生成推荐电影清单返回给网页进行网页的生成该用户的定制首页。

用户播放电影时，显示当前播放电影相似的推荐电影。

新用户登录时，根据电影热度排行数据生产默认推荐电影列表

推荐过程如图5.2所示



图5.2推荐功能流程图

5.2.4 管理员功能

电影信息管理：电影信息的管理操作。

电影资源管理：电影播放视频资源链接、图片的资源的管理。

电影播放热度统计：分析电影播放记录信息表，对电影播放次数、评分等信息进行统计、并且依照这些信息计算出电影的热度，结合前端实现播放数据的柱形图绘制。

用户信息、行为查询：管理员对网站平台用户的信息和行为进行查询、管理，贯彻用户优先原则，管理员无法修改用户信息。

## 5.3 数据库设计

由于Python存在字典这一数据结构，转化成二进制编码的JSON数据成本低，且与MongoDB数据库使用的BSON数据格式具有较好的兼容性，同时与前端使用的数据格式支持较好，减少前端、服务端、数据库之间的数据格式转化成本，易于在网络传输存储，存储过程灵活，不受关系型数据库的限制，对于1对多的实体可以用集合对象存储。在性能方面，以二进制文件存储的MongoDB具有较快的传输速度，利用C语言编写的底层对于文档的解码编码也是非常迅速的。

### 5.3.1 用户数据表

用户表数据实体如图5.3所示，用户数据表字段名、含义、数据类型如表5-1所示。其中\_id是MongoDB数据库自动生产的记录表示序号。



图5.3 用户数据实体图

表5-1 用户数据表数据字段

|  |  |  |  |
| --- | --- | --- | --- |
| 用户数据表 | | | |
| 序号 | 数据库字段名 | 中文含义 | 字段类型 |
| 1 | userId | 用户编号 | int |
| 2 | username | 用户名 | string |
| 3 | name | 真实姓名 | string |
| 4 | pwd | 密码 | string |
| 5 | phone | 电话 | string |
| 6 | email | 邮箱 | string |
| 7 | other | 备注 | string |
| 8 | status | 删除标志 | int |
| 9 | \_id | MongoDB标识符 | ObjectId |

### 5.3.2 电影数据表

电影数据表实体如图5.4所示，用户数据表字段名、含义、数据类型如表5-2所示



图5.4电影数据实体

表5-2 电影数据表数据字段

|  |  |  |  |
| --- | --- | --- | --- |
| 电影数据表 | | | |
| 序号 | 数据库字段名 | 中文含义 | 字段类型 |
| 1 | movieId | 电影编号 | int |
| 2 | title | 电影名 | string |
| 3 | time | 上映时间 | string |
| 4 | genres | 电影类别 | string |
| 5 | describe | 电影介绍 | string |
| 6 | imdbId | 详情链接 | string |
| 7 | vide\_url | 视频资源链接 | String |
| 8 | img\_url | 图片链接 | String |
| 9 | status | 删除标志 | int |
| 10 | \_id | MongoDB标识符 | ObjectId |

### 5.3.3 电影播放记录表

播放记录数据表实体如图5-5所示，该表存在推荐系统需要使用的训练数据字段为用户编号、电影编号、评分，播放记录字段详情如表5-3所示



图5.5 播放记录数据实体

表5-3 播放记录数据字段

|  |  |  |  |
| --- | --- | --- | --- |
| 播放记录表 | | | |
| 序号 | 数据库字段名 | 中文含义 | 字段类型 |
| 1 | userId | 用户编号 | int |
| 2 | movieId | 电影编号 | int |
| 3 | ratings | 评分 | int |
| 4 | title | 电影名 | string |
| 5 | comment | 评价 | String |
| 6 | status | 删除标志 | int |
| 7 | \_id | MongoDB标识符 | ObjectId |

### 5.3.4 缓存数据表

为了提高系统运行效率，使用缓存数据表来提高效率

1、电影相似性矩阵表，是相似性算法中的临时数据存储表，存储电影之间的相似性集合对象，数据实体如图5.6，电影相似性矩阵数据字段，如表5-4。



图5.6 电影相似性矩阵数据实体

表5-4 电影相似性矩阵数据字段表

|  |  |  |  |
| --- | --- | --- | --- |
| 电影相似性矩阵表 | | | |
| 序号 | 数据库字段名 | 中文含义 | 字段类型 |
| 1 | movieId | 电影编号 | int |
| 2 | simi\_movie | 相似电影 | Object |
| 3 | status | 删除标志 | int |
| 4 | create\_time | 生成时间 | date |
| 5 | \_id | mongodb标识符 | ObjectId |

2、用户相似性矩阵表数据表实体如图5.7所示，临时保存用户相似性数据，以便推荐系统调用，用户相似性矩阵数据字段如表5-5所示



图5.7 用户相似性矩阵表数据表实体

表5-5 用户相似性矩阵表数据表数据字段

|  |  |  |  |
| --- | --- | --- | --- |
| 用户相似性矩阵表 | | | |
| 序号 | 数据库字段名 | 中文含义 | 字段类型 |
| 1 | userId | 用户编号 | int |
| 2 | simi\_user | 相似用户 | Object |
| 3 | status | 删除标志 | int |
| 4 | create\_time | 生成时间 | date |
| 5 | \_id | mongodb标识符 | ObjectId |

3、用户推荐电影表数据表实体如图5.8所示，推荐算法中生成的临时推荐信息缓存表，为了提高推荐系统效率而存在，推荐电影缓存表定时更新。用户推荐电影表数据表数据字段，如表5-6所示。



图5.8 用户推荐电影表实体

表5-6 用户推荐电影表数据字段

|  |  |  |  |
| --- | --- | --- | --- |
| 用户推荐电影表 | | | |
| 序号 | 数据库字段名 | 中文含义 | 字段类型 |
| 1 | userId | 用户编号 | int |
| 2 | rec\_movie | 推荐电影 | Object |
| 3 | create\_time | 生成时间 | date |
| 4 | status | 删除标志 | int |
| 5 | \_id | MongoDB标识符 | ObjectId |

4、电影播放次数统计表数据表实体如图5.9所示，统计电影热度的临时缓存表，根据电影播放热度决定排行，可以作为缺省用户默认途径列表，电影播放次数统计表数据字段，如表5-7所示。



图5.9 电影播放次数统计表数据实体

表5-7 电影播放次数统计表数据字段

|  |  |  |  |
| --- | --- | --- | --- |
| 电影播放次数统计表 | | | |
| 序号 | 数据库字段名 | 中文含义 | 字段类型 |
| 1 | movieId | 用户编号 | int |
| 2 | title | 电影名 | string |
| 3 | count | 电影播放次数 | int |
| 4 | Ratings | 总评分 | int |
| 5 | create\_time | 生成时间 | date |
| 6 | status | 删除标志 | int |
| 7 | \_id | MongoDB标识符 | ObjectId |

## 5.4 本章小结

本章介绍个性化视频网站开发过程中使用的技术、需求分析，功能概要设计以及MongoDB数据库下的文档数据表设计,制定了项目开发需求的计划。在本章完善了前期需求准备工作，建立数据流通标准，为下章功能设计奠定基础，网站中采用两种协同过滤算法的混合推荐系统为用户推荐电影。明确项目功能需求，具有用户观看电影、评分、评论的功能，后台管理，数据可视化的功能。进行数据库设计和搭建，生成项目所需初始化据。对于数据库的搭建使用虚拟机下搭建MongoDB和redis在局域网内，可以模拟真实的服务端，数据端的分离过程。

1. 个性化视频推荐网站的实现

系统实现是对需求分析中的功能模块的基本实现，主要过程有后台功能实现，前端页面和脚本功能的实现，包含数据库配置、系统设计、代码架构设计和模块设计。

## 6.1 开发环境搭建

通过综合考虑推荐系统实现的方式，引出项目开发环境、技术选择。

|  |  |  |
| --- | --- | --- |
| 开发工具 | 开发语言 | Python |
|  | 后端框架 | Flask |
|  | 前端框架 | JinJa2、Bootstrap |
|  | 数据库 | MongoDB |
| 支持工具 | 数据验证 | 正则表达式、MD5加密算法 |
|  | 缓存数据库 | Redis |
|  | 数据刷新展示 | Ajax、canvas等 |

## 6.2 用户网站功能实现

### 6.2.1 电影视频播放网站网页设计清单

综合项目的功能需求，制定项目需要的网页数量和功能

|  |  |
| --- | --- |
| 登录网页 | 登录页面 |
|  | 注册页面 |
|  | 修改密码页面 |
|  | 修改用户信息 |
| 主页 | 个人首页 |
|  | 播放页面 |
|  | 搜索页面 |
|  | 查找页面 |
|  | 播放记录页面 |
| 后台管理 | 用户信息管理 |
|  | 电影信息管理 |
|  | 电影信息统计 |
|  | 电影资源管理 |

### 6.2.2 用户个人信息管理

用户登录、用户注册等功能，由于HTTP协议的无状态性，所以采用Session和Cookies来进行记录用户的登录状态。为防止Cookies欺骗、采用Token令牌验证方式，使用MD5加密算法来生成Token令牌，服务器存储在Redis缓存中，客户端存储在Cookie中。

具体方法为：使用Ptyhon中的hashlib模块，利用用户名、密码、以及加盐固定字符串和登录时间生成用户token令牌信息，保存在Cookes和服务端的Session和Redis服务器中，token过期时间设置为3小时，用户退出登录时删除token。当用户登陆时，检测Cookies中携带的token令牌与Session中对照，当Session中不存在时，从Redis数据库中读取token令牌验证，网站登录界面如图6.1所示，网站的注册页面如图6.2所示。



图6.1 网站登录页面展示图



图6.2 网站注册页面展示图

修改信息功能实现：允许用户修改个人信息、登录密码等，前端实现用户信息格式验证，后端对传输的数据进行信息格式验证、数据存在性验证、数据安全性验证。密码修改功能如图6.3所示。

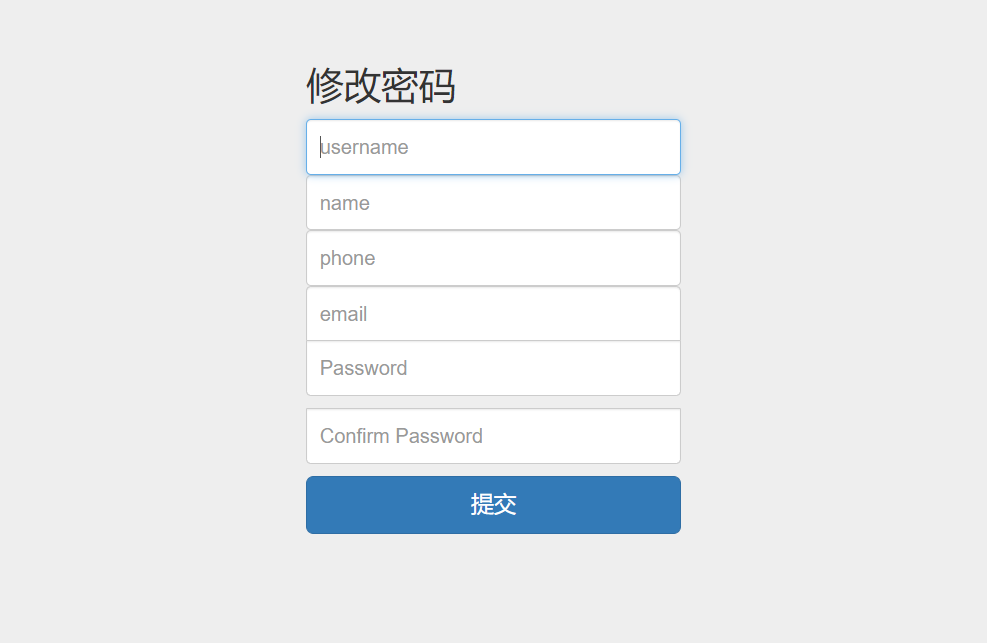


图6.3 修改密码页面展示图

播放记录管理：用户删除个人的播放记录，只允许用户删除播放及评分记录，后端采用伪删除实现，将数据标记为过时数据，播放记录管理功能如图6.4所示。

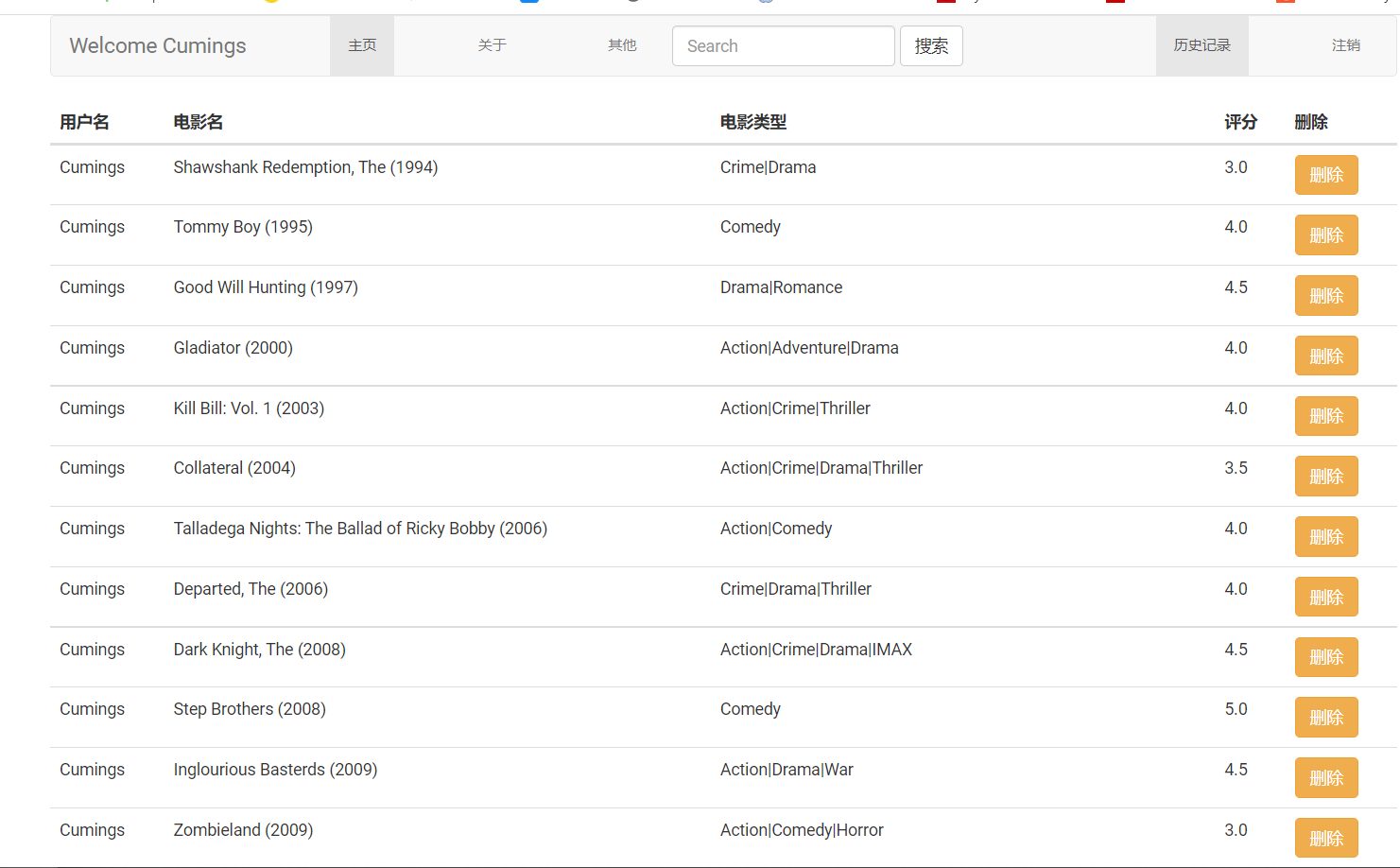


图6.4 播放及评分记录管理页面展示图

### 6.2.3 用户操作行为功能

用户播放功能，用户可以查看视频信息、播放视频的功能

用户评论功能，用户可以在电影下方的评论区发表、查看评论的功能

用户评分功能，用户对于电影的详细信息、简介查看，对电影进行打分

功能展示如图6.5所示。



图6.5播放视频页面展示图

### 6.2.4 站内搜索功能

全局搜索，解除了用户查询条件限制，用户可以通过电影名、电影类型、上映时间等进行精确或者模糊查询，利用MongoDB文档型数据，利用查询条件在所有符合字段里面进行查询。查询实现方式利用用户传入的条件组成正则表达式，在数据库中对符合字段进行匹配，例如电影名、类型、描述等字段。搜索结果展示界面如图6.6所示。

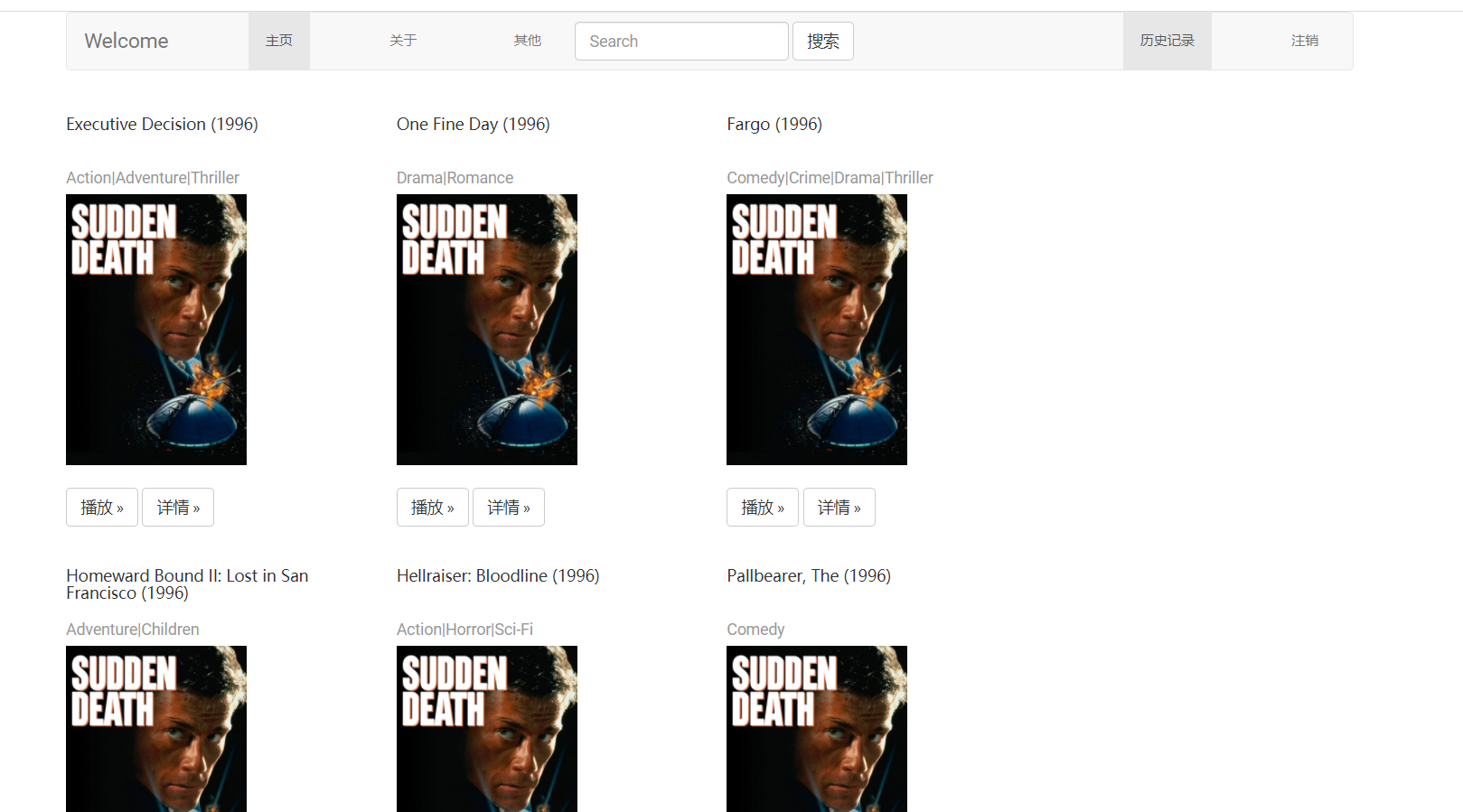


图6.6 搜索结果展示图

### 6.2.5 推荐功能

用户喜好电影推荐，用户进入网站首页时，系统会给出个性化首页，个性化的个人首页如图6.7所示。

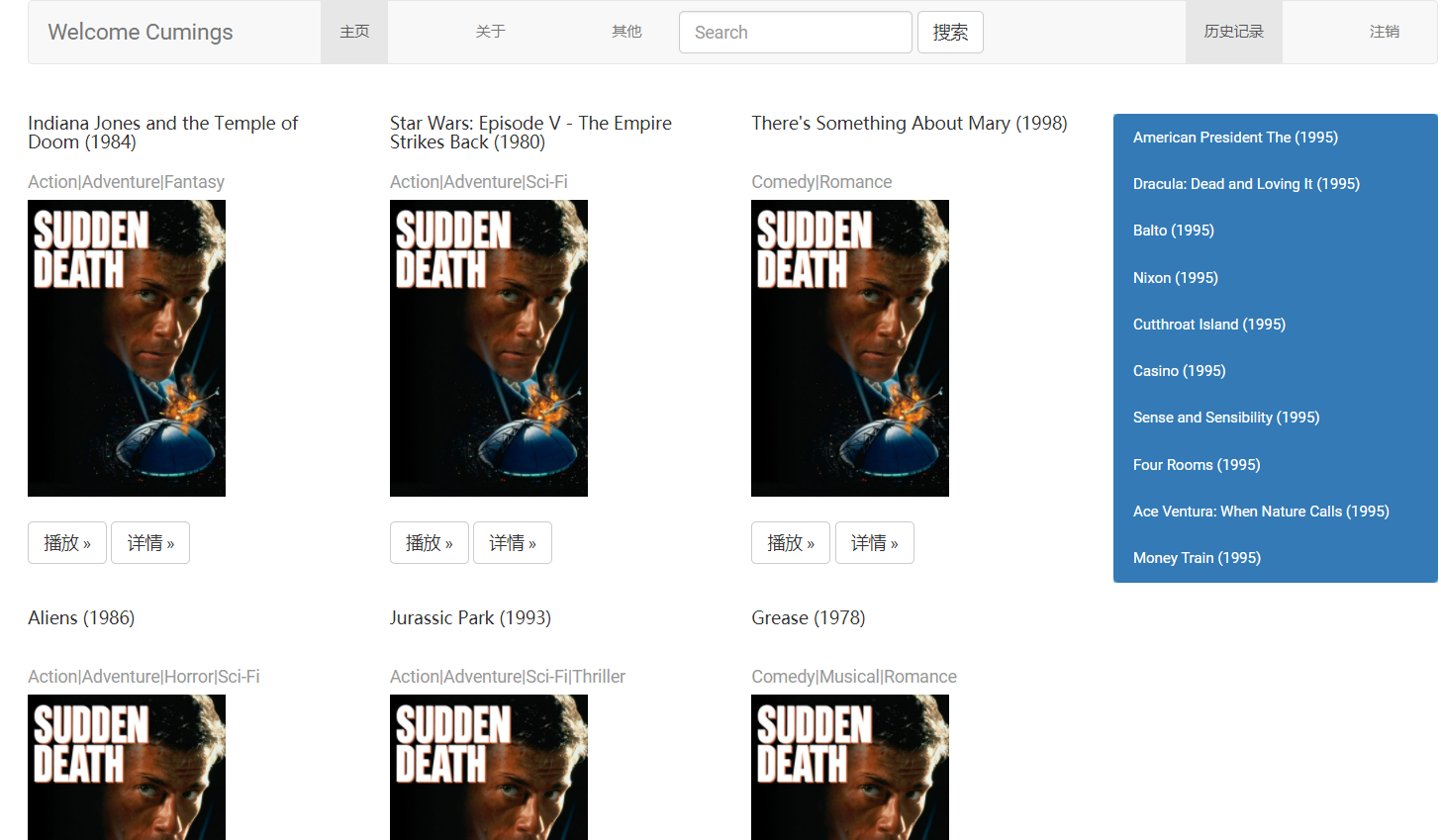


图6.7 用户定制的个人首页展示图

## 6.3 后台管理功能实现

利用ajax实现异步操作，删除操作全部采用伪删除，对于数据修改增加前端、后台的验证，验证规则需要单独的规定，分页功能实现具有一定的难点。同时，利用Jinja2的前端网页继承，利用一个通用的页面父模板，减少每个网页的编写工作量。

### 6.3.1 电影信息管理

对于电影信息的管理、修改功能，展示如图6.8所示。

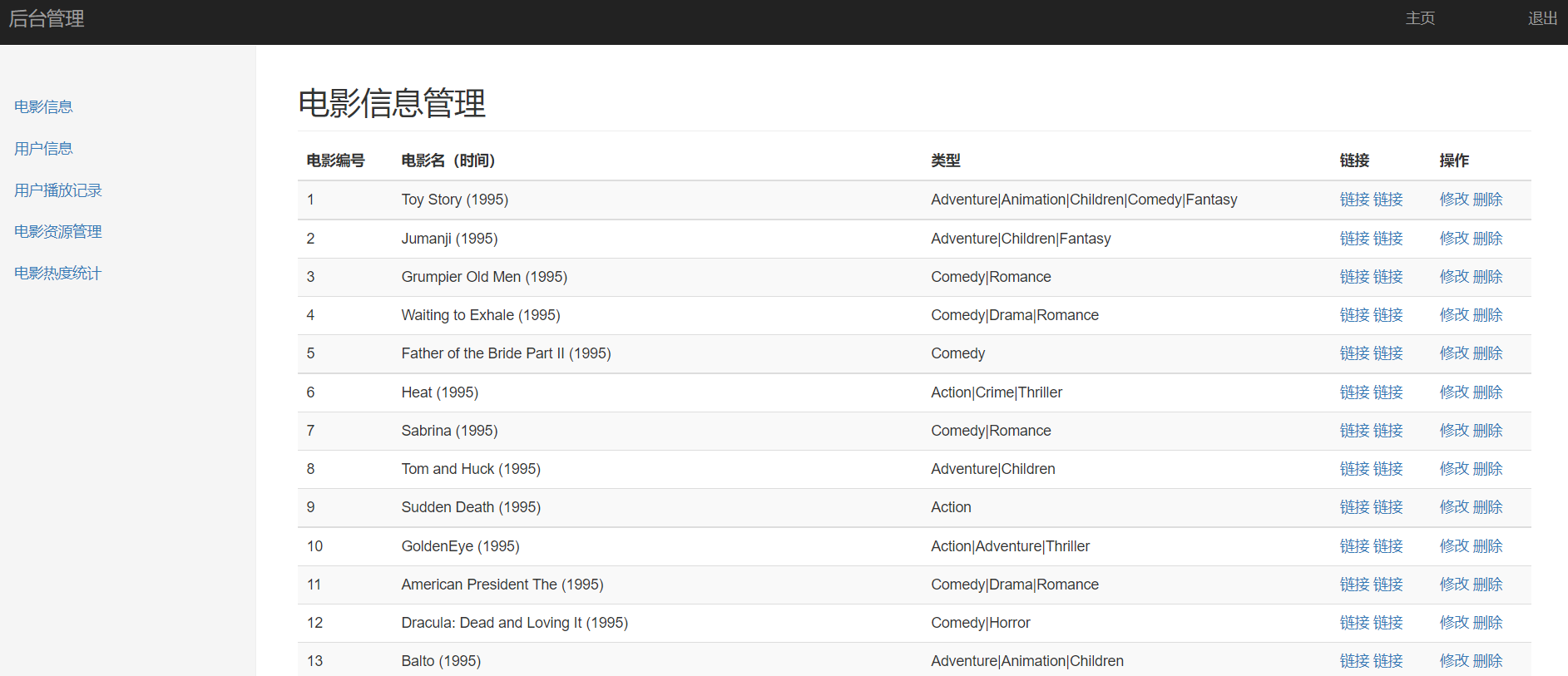


图6.8 电影信息管理功能展示图

### 6.3.2 电影资源管理

电影播放视频资源链接、图片的资源的管理，功能的具体展示效果如图6.9所示。



图6.9 电影资源链接管理展示图

### 6.3.3 电影播放热度统计

分析电影播放记录信息表，对电影播放次数、评分等信息进行统计，功能展示如图6.10所示。

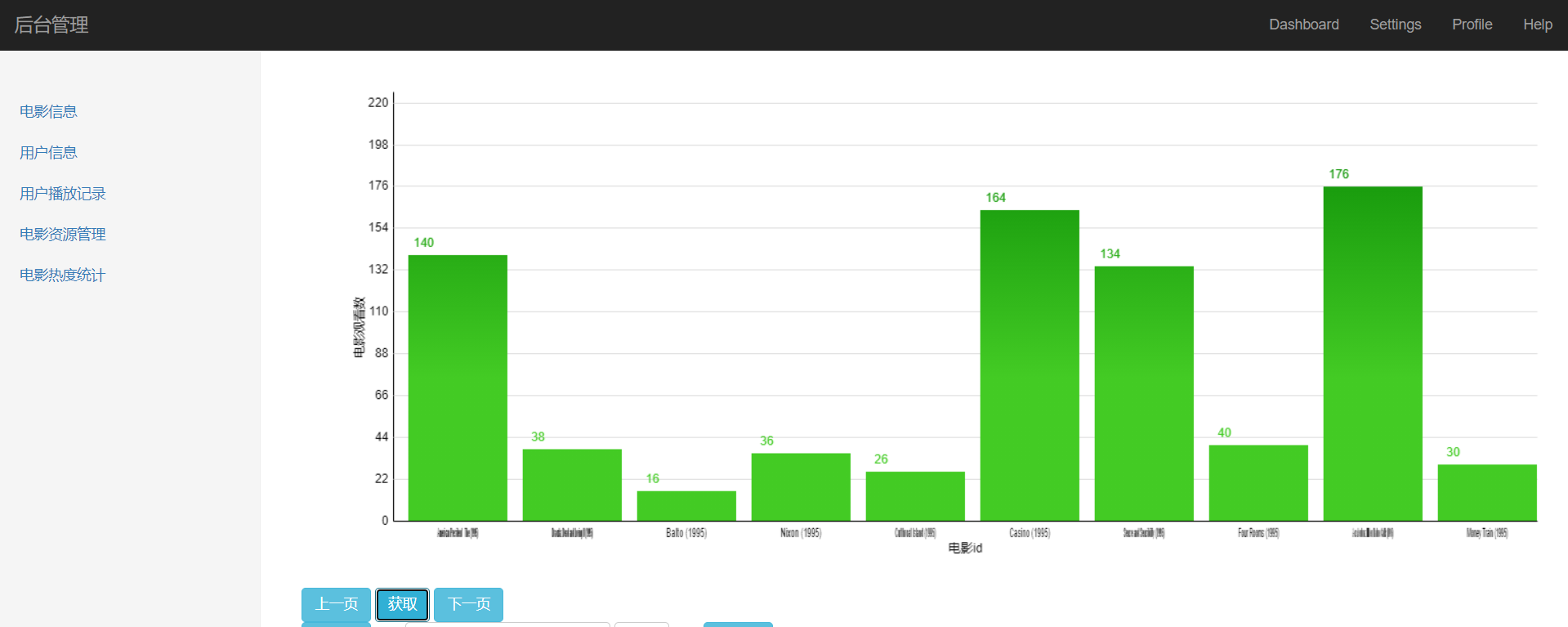


图6.10 电影热度数据可视化展示图

### 6.3.4 用户信息管理

管理员对用户的信息、播放记录查询，为保护用户隐私，只允许查询操作，具体展示效果如图6.11所示。

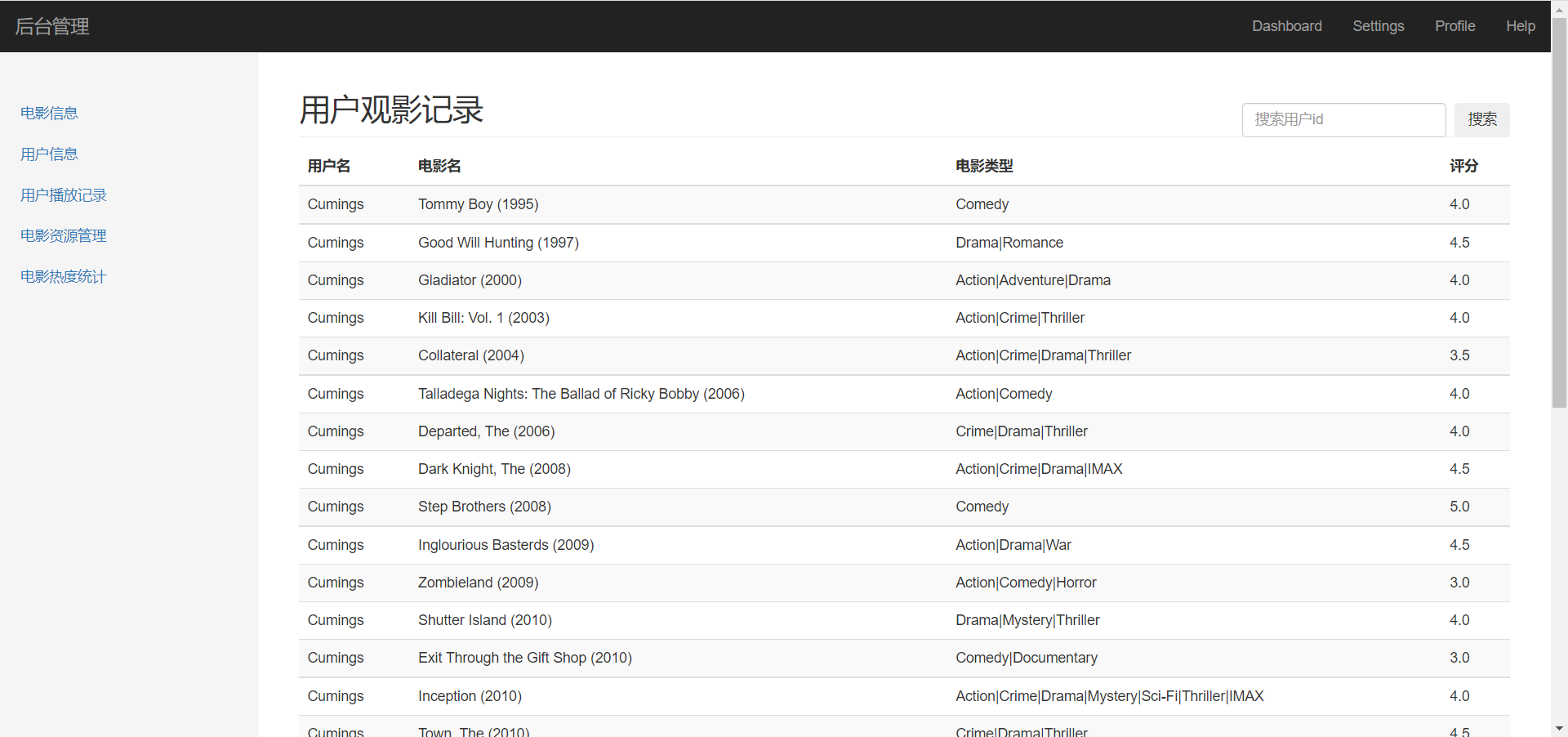


图6.11 用户观影及评分记录管理页面展示图

## 6.4 网站前端首页功能实现

### 6.4.1 前端柱形图绘制

前端使用js操作canvas画布实现数据的可视化展示，使用ajax异步请求实现柱形图的更新，可以直观地查看电影热度、播放次数，柱形图绘制效果如图6.12所示。

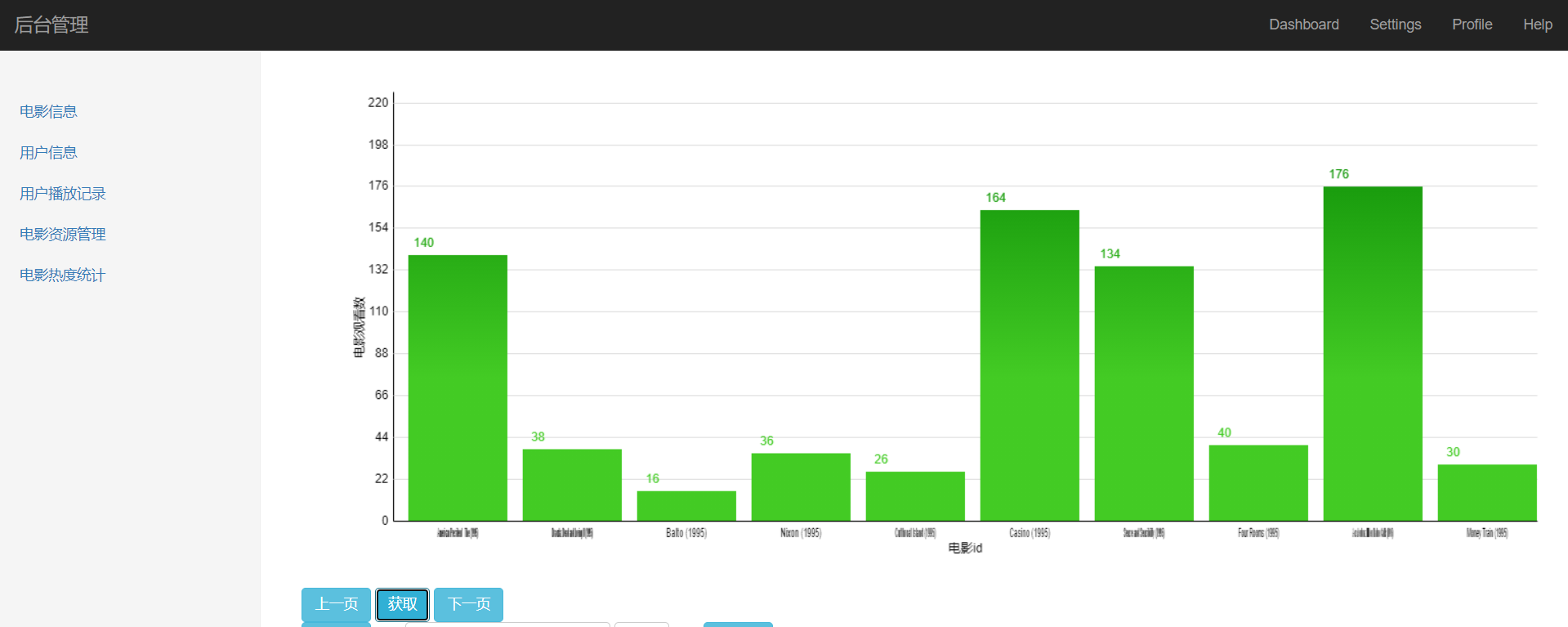


图6.12 电影热度数据可视化展示图

### 6.4.2 前端分页功能实现

当数据量达到一定级别时，需要进行数据切割，否则会导致传输速度缓慢，页面响应迟钝，影响用户体验。按照分页的操作角色分为数据库分页、服务端分页、前端分页。

1、数据库分页，利用数据库分页功能进行分页，部分读取数据并传输，需要传入读取记录的起点和读取的条数。该方法优点在于不会造成内存溢出，但由于依靠数据库实现分页，翻页的速度比较慢，对数据库访问压力增大。

2、服务端分页，一次性将所有的数据放置在服务器内存之中，由服务端代码进行分页读取。该方法分页速度快，对于数据库压力小，但是由于数据缓存在服务器内存中，容易造成内存溢出。

3、前端分页，将所有数据一次性加载到前端实现分页操作，该方法虽然请求次数少，但首次传输缓慢、页面加载迟钝，数据更新不及时，安全性低。

本设计项目访问量较少，故采用数据库分页方式速度较快、实现简单。

## 6.5 本章小结

本章主要对系统网站功能模块进行实现，对于实现方案的验证和思考，通过运行测试展示检测网站系统功能的完善性，针对项目中一些特殊情况的处理，进行单独的分析和实现，在项目测试过程中发现问题修改问题。

结 论

在推荐算法领域，协同过滤作为推荐系统的基础组成算法之一被广泛应用在亚马逊、YouTube等电商、视频网站中。传统的协同过滤存在的缺陷，通过许多学者不断改进研究得到提升，针对冷启动、稀疏数据等问题拥有了较好的解决方案，利用混合推荐算法来扬长避短是推荐系统的发展主流方向。

本设计的主要工作是结合协同过滤进行实际应用。对于以余弦相似度、杰卡德系数为核心的协同过滤推荐算法进行分析，确定最终采用的方法。在应用推荐系统到个性化视频网站过程中，对推荐系统进行改进。根据系统功能需求进行主要模块功能的详细设计及项目数据库表的设计，确立大致的实现方向和思路。利用NoSql数据库缓存推荐系统的推荐结果，调整推荐系统运行时机和次数提高推荐系统的效率。采用模块化的设计方法，分别设计每个功能并绘制主要功能推荐系统的流程图，探究推荐系统工作原理，同时搭建测试环境，基本实现各个模块功能。

推荐领域的算法有着非常广阔的研究空间，目前推荐系统大多数依赖于获取用户的信息和行为数据，因此该过程涉及大量用户的隐私，在此过程中容易触发侵犯用户隐私的问题，在互联网中，保护用户的隐私是一个比较重要的问题。

在推荐系统中，涉及到用户都是数以亿计，如何提高推荐系统的效率是迫切需要研究的方向，利用分布式平台高效、准确地计算用户日志中的行为数据，建立一个完善的推荐模型。

参考文献

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

[2]王立才, 孟祥武, 张玉洁. 上下文感知推荐系统[J]. 软件学报, 2012(01):1-20.

[3]刘建国, 周涛, 郭强,等. 个性化推荐系统评价方法综述[J]. 复杂系统与复杂性科学, 2009(03):1-10.

[4]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.

[5]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(002):350-362.

[6]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, 019(001):1-15.

[7]王敏. 基于协同过滤的电子商务个性化推荐系统设计和实现[D]. 广州大学, 2019.

[8]贾忠涛. 基于协同过滤算法的电影个性化推荐系统设计与实现[J]. 软件导刊, 2015, 14(1):86-86.

[9]孙广东. 基于协同过滤的图书推荐系统设计与实现[J]. 2017.

[10]迟轩. 基于协同过滤算法的推荐系统框架设计与实现[D].上海交通大学,2013.

[11]尹作文. 基于混合协同过滤的电子商务推荐系统的研究与应用[D]. 武汉理工大学, 2015.

[12]彭兴. 基于Hadoop的电子商务推荐系统的设计与实现[D]. 闽南师范大学, 2016.

[13]孟琳萱. 基于协同过滤的电商个性化推荐系统的设计与实现[J]. 数字化用户, 2019, 025(032):111,113.

[14]迟轩. 基于协同过滤算法的推荐系统框架设计与实现[D]. 上海交通大学.

[15]尹作文. 基于混合协同过滤的电子商务推荐系统的研究与应用[D].

武汉理工大学.

[16]朱小强, 张琳. 一种改进的协同过滤推荐算法[J]. 现代计算机:中旬刊, 2015.

[17]董辉, 方晓, 马健,等. 基于情景感知的移动电商用户—项目聚类协同过滤推荐算法[J]. 广西民族大学学报(自然科学版), 2018, 024(002):61-68.

[18]电子商务. 基于模糊推理的web客户需求协同过滤推荐算法. 2011.

[19]李亚欣. 基于协同过滤的电子商务个性化推荐算法研究[D]. 东北财经大学, 2012.

[20]刘庆鹏. 基于协同过滤的电子商务个性化推荐算法研究[D]. 海南大学, 2012.

[21]贺桂和. 基于用户偏好挖掘的电子商务协同过滤推荐算法研究[J]. 情报科学, 2013, 031(012):38-42.

[22]王纪辉, 赵卓宁. 基于协同过滤算法的电子商务网站个性化推荐系统设计[J]. 成都信息工程学院学报, 2007, 22(0z1):161-167.

[23]周春华, 沈建京, 李艳,等. 经典推荐算法研究综述[J]. 计算机科学与应用, 2019, 009(009):P.1803-1813.

[24]赵森. 融合用户偏好和项目关联的协同过滤算法研究[D].吉林大学,2020.

[25]周春荣.百度团购个性化推荐引擎的设计与实现[D].北京交通大学.2015年

[26]张洋.推荐系统中冷启动问题与评估方法研究[D].华北电力大学(北京).2017年

[27]陈艺婷.基于混合协同过滤的电商推荐系统的设计与实现.南京邮电大学.2020年

[28]Bouzid M , Bonnefoy D , Lhuillier N , et al. RECOMMENDER SYSTEM[J]. US.

[29]Sarwar B M . Application of Dimensionality Reduction in a Recommender System : A Case Study[C]// Acm Webkdd Web Mining for E-commerce Workshop. 2000.

[30]Hema B , Shikha M . A Multi -Perspective Evaluation of MA and GA for Collaborative Filtering Recommender System[J]. International Journal of Computer Science & Information Technology, 2010, 2(5):103-122.

致 谢

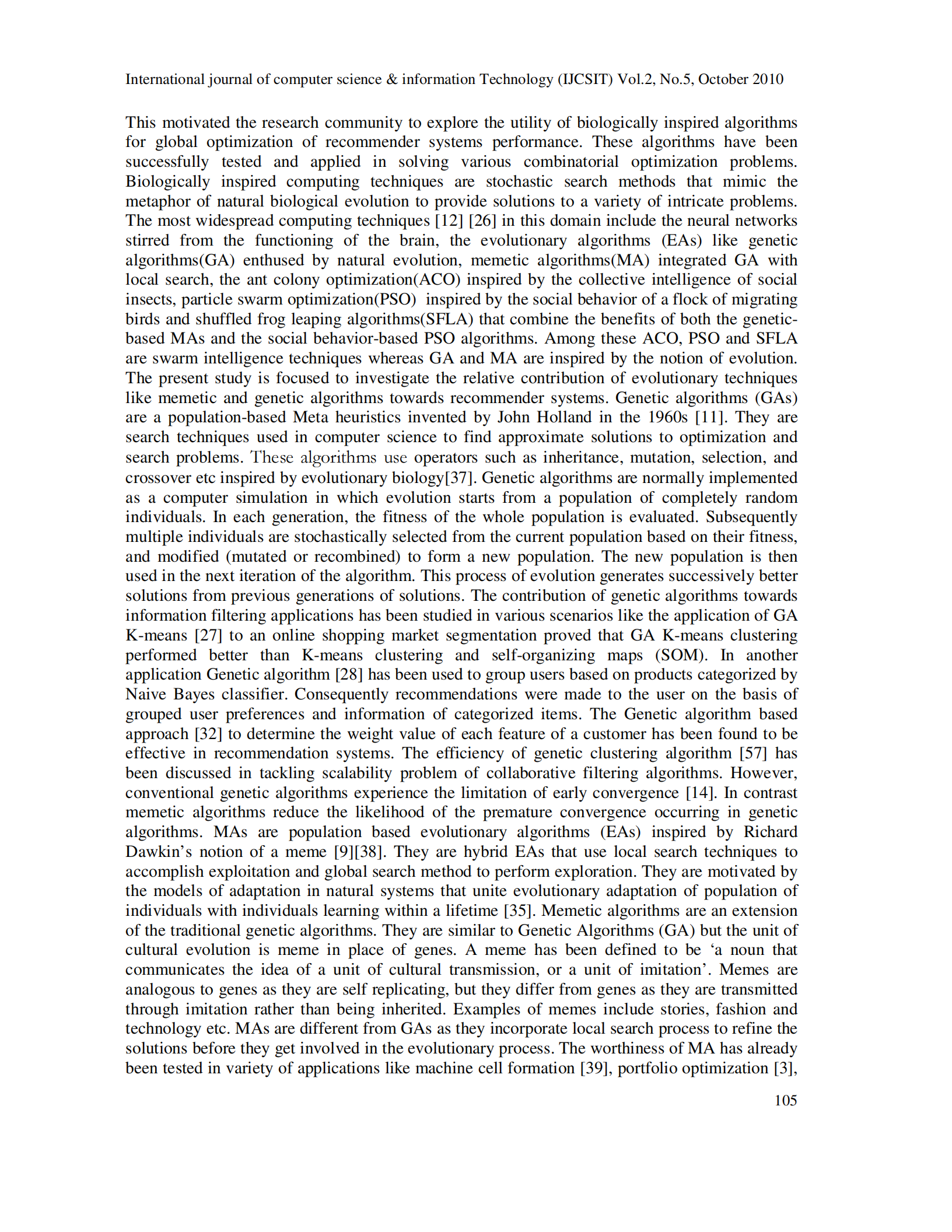
本论文攥写期间遇到很多或大或小的问题，在解决问题的过程深感自己的不足，毕业不是学习的终点也是学习的起点，在解决论文的问题过程中，感谢老师同学之间的帮助，在各位的帮助之下，明确了论文的思路，制定了论文的大纲，在毕设的设计过程中，感谢老师给予的解答，或许老师对于新技术的理解不足，但是宝贵的经验仍然是难得可贵的帮助。

时光荏苒，流年易逝，不知不觉中已经来到毕业，回想四年经历，也算是丰富多彩了，在学习生活中，感谢同学老师给予的帮助，在四年成长历程中、收获了很多，人生漫漫，有还有二十多个四年，希望在未来的时间里，可以砥砺前行、不负青春。

附 录







摘要

进化算法在解决复杂问题中的日益普及，激发了研究人员探索其在推荐系统中的应用。推荐系统是一种智能的web应用程序，它根据用户的陈述和未陈述的需求生成推荐。遗传算法和模因算法等进化算法被认为是最成功的组合优化方法之一。模因算法（Memetic Algorithms，MAs）是一种改进的遗传算法，它在进化方案中引入了局部搜索。每一代后对每个解的局部搜索过程有助于提高MA的收敛速度。本文对模因进化算法和遗传进化算法在基于模型的协同过滤推荐系统中的应用进行了多角度的比较研究。在MovieLens数据集上对模因算法和遗传算法的决策支持和统计效率进行了实验研究。从聚类数的变化、增加用户数的影响、不同的推荐数以及使用一个或多个聚类来计算未评分项目的评分等不同角度对算法进行了分析。结果表明，从各个角度看，模因协同过滤算法比遗传协同过滤算法具有更好的预测精度。

关键词

模因协同过滤，遗传协同过滤，模因推荐系统，遗传推荐系统，进化协同过滤。

介绍

万维网上信息的爆炸式增长导致了智能网络应用的发展。这些应用程序考虑了资源的异构性、用户兴趣和移动性，以智能地利用web上大量、丰富和共享的web资源和服务。这类web应用的例子包括web个性化、web挖掘、语义web和智能代理等。然而，近年来，被称为推荐系统的自适应/个性化web应用变得越来越流行。这些系统试图识别活跃用户的特征，然后根据用户的个人历史或朋友、亲戚和同龄人等的意见提供建议，以方便他们做出决策。在用于开发推荐系统的各种技术中，协作过滤、基于内容的过滤和混合技术更为常见[1]。其他过滤技术包括人口统计过滤，它使用用户的年龄、性别、资历等人口统计信息来提出建议。基于效用的推荐系统根据项目对用户的有用性来推荐项目。基于知识的推荐系统利用特定商品如何满足特定用户需求的知识来进行推荐。最近，一些作者提出了混合的方法来解决这个问题

克服了使用任何单一方法的缺点。这些系统称为混合推荐系统[17][56]。所有这些方法在描述用户概要和信息产品的方式上都有所不同，并将其与生成建议进行比较，例如在基于内容的过滤中；建议根据用户以前的首选项进行项目。另一方面，协同过滤（CF）[47]技术主要通过比较活动用户的配置文件和其他用户来识别活动用户的社区，然后根据社区偏好推荐项目。通常，协作过滤中的概要文件由用户对各种感兴趣项目所做的显式/隐式评级表示。显式评级/投票是指用户对产品表示的偏好，通常是离散数值尺度上的。例如，GroupLens系统使用1到5的比例对Netnews文章进行评级。用户在阅读完文章后明确地对每篇文章进行评分。隐式评级是指根据网络浏览数据或购买历史等对用户行为或选择进行解释，输入投票或偏好。协作过滤不利用产品特性的任何信息，因此已成功应用于广泛的应用[18]。随着织锦系统的发展，协同过滤技术的研究蓬勃发展

1. 这是基于基于拉的协同过滤方法。在这种系统中，愿意提出建议的用户必须主动地从数据库中提取建议。随着该系统的巨大成功，其他研究人员开始为推式协同过滤系统的发展做出贡献[33]。与基于拉动的系统相比，这些系统的用户有权将其兴趣项推给朋友和同龄人。随着半自动系统的广泛接受，自动协同过滤系统应运而生，如在Usenet新闻组文章领域开发的GroupLens system[30][45]，Bellcore的视频推荐器[20]推荐电影和Ringo system[48]用于音乐等，其他示例包括Amazon.com for books，Jester system[16]用于推荐笑话和PHOAKS系统[49]，以帮助用户在web上找到相关信息。迄今为止，学术界和业界在不同应用领域开发的各种推荐系统的预期统计数字[43]大约为21个，用于推荐Web、13个用于电影、11个用于新闻、10个用于文档和信息、6个用于音乐、5个用于信息过滤和共享等。其他领域，如旅游、电子商务、网上有趣的变化等都有1-3个推荐系统。然而，这些数字与WWW的规模相比太小，一些应用领域，如健康、教育和地理等，仍有待探索。

文献[7]从基于模型的协同过滤和基于记忆的协同过滤两个角度探讨了协同过滤技术，基于记忆的协同过滤系统使用整个用户项目评分数据集进行预测。这些系统使用统计技术来识别一组被称为邻居的用户，这些用户的过去行为与目标用户相似。然后结合邻居的偏好来计算对目标用户的推荐。相反，基于模型的CF系统将完整的数据集分为源数据集和目标数据集。源数据集用于使用各种机器学习算法（如贝叶斯网络[24]、聚类算法[29]等）训练模型。此后，训练后的模型用于为目标数据集中的活动用户生成建议。将用户分组有助于提高推荐系统的准确性，因此聚类算法在创建相似购买用户的社区/类别时变得更加普遍。现有的聚类算法，如K-means算法，虽然简单，聚类时间短，但很有可能陷入局部最优。此外，这些技术表现出不一致的性能[51]，这可能是因为数据通常非常稀疏，也可能是因为人们的品味将数据分为多个类别。由于这些传统的聚类方法都是即席的，而且改进的方法也很模糊，因此不能简单地利用所有可访问的知识。