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

系部名称： 计算机科学与技术学院

专业班级： 计算机科学与技术17-4

学生姓名： 陈盈涛

指导教师： 杨泽雪

职 称： 副教授

**黑 龙 江 工 程 学 院**

二○二一年六月

论文封面示例，同前

本科学生毕业论文

基于协同推荐的视频网站的设计与实现

系部名称： 计算机科学与技术学院

专业班级： 计算机科学与技术17-4

学生姓名： 陈盈涛

指导教师： 杨泽雪

职 称： 副教授

**黑 龙 江 工 程 学 院**

二○二一年六月

外文封面示例Times New Roman体，字号与中文对应

The Graduation Design for Bachelor's Degree

Design of Construction Machinery

Hydraulic Systems

Candidate：Chen Yingtao

Specialty：Computer Science And Technology

Class：17-4

Supervisor：Associate Prof. Yang Zexue

Heilongjiang Institute of Technology

2021-06·Harbin

外文封面示例Times New Roman体，字号与中文对应

The Graduation Thesis for Bachelor's Degree

Study on Construction Machinery

Hydraulic Systems

Candidate：Zhang Xianfen

Specialty：Construction Machinery

Class：B02-20

Supervisor：Associate Prof. Li Chunxiang

Heilongjiang Institute of Technology

2006-06·Harbin

中文摘要示例

摘□□要

（小二号黑体字间加2空格，上下各空一行）

本论文是在社会主义市场经济理论的指导下，通过对黑龙江省道路货运市场现状的分析，根据中心城市所处的地位和货运市场的特点，探讨中心城市货运市场建设的发展模式，设计了货运中介服务组织的组织形式及信息传递方式，进而提出货运场站建设的目标及其评价指标体系……

（宋体小四号字 1.35倍行距）

关键词：道路；货运；市场；模式；系统

（小四号黑体） （小四号宋体）

Ⅰ

**ABSTRACT**

外文摘要示例

（小二号Times New Roman体，加黑，上下各空一行）

Under the direction of socialist market economic theory,through the analysis of current situations of highway freight transport market of Heilongjiang province, this thesis discusses the developing patterns and the organization forms of intermediary service of freight transport and transferring ways of the related information and provides the construction objects and the index system of building freight transport yard, according to the position of central city and the characteristics of its freight market.

（Times New Roman体，字号、行间距与中文相同）

**Key words**: Highway；Freight Transport；Market；Pattern；System

（加黑）

Ⅱ

目□□录

目录示例

（小二号黑体，字间加2空格，上下各空一行）

摘要………………………………………………………………………………………Ⅰ

Abstract ………………………………………………………………………………Ⅱ

第1章 绪论………………………………………………………………………………1

□□1.1 概述………………………………………………………………………………2

（章的名称四号黑体，其余小四号宋体）

·

第2章 发动机气缸的磨损…………………………………………………………20

2.1 活塞-缸套振动机理……………………………………………………………22

2.1.1 测点布置…………………………………………………………………22

·

2.2 本章小结…………………………………………………………………………26

（章的名称四号黑体，其余小四号宋体）

·

·

结论………………………………………………………………………………………33

参考文献 ………………………………………………………………………………36

致谢………………………………………………………………………………………38

附录………………………………………………………………………………………40

（四号黑体）

目 录

[第一章 绪论 1](#_Toc7703)

[1.1 研究背景 1](#_Toc13653)

[1.2 研究意义 2](#_Toc6926)

[1.3 研究目的 2](#_Toc26233)

[1.4 本文安排 3](#_Toc20898)

[第二章 研究现状及分析 4](#_Toc7767)

[2.1 推荐算法的发展 4](#_Toc27227)

[2.1 推荐算法的研究现状 4](#_Toc6616)

[2.2 国内外推荐算法应用现状 5](#_Toc4851)

[第三章 推荐算法以及相关介绍 6](#_Toc2658)

[3.1推荐算法总览 6](#_Toc20354)

[3.2 相似度算法总览 11](#_Toc17099)

[3.3 推荐系统评测 12](#_Toc4642)

[第四章 协同过滤推荐算法实验与应用 14](#_Toc5372)

[4.1 协同过滤推荐算法实验设计 14](#_Toc13441)

[4.2 协同过滤推荐算法分析 14](#_Toc11688)

[4.3 基于协同过滤推荐算法的推荐系统评测 14](#_Toc2879)

[第五章 电影推荐系统的实现 15](#_Toc26306)

[5.1 总体设计环节 15](#_Toc18704)

[5.2 数据库方面 15](#_Toc3734)

[5.3 算法采用 15](#_Toc22505)

[5.4 启动数据 15](#_Toc20653)

[5.5展示界面 15](#_Toc16597)

[第六章 总结与展望 16](#_Toc5871)

[6.1论文工作总结 16](#_Toc16156)

[6.2 论文展望 16](#_Toc11413)

[第七章 参考论文 17](#_Toc16859)

第1章 绪□□论

正文示例

（小二号黑体居中，上下各空一行）

## 1.1 概述 （四号黑体）

□□本文以EQ6100发动机为实验对象…………（小四号宋体，1.35倍行距）

……

1.1.1 活塞环径向振动固有频率（小四号黑体）

□□通过实验对本文所提出的理论及技术做进一步…………

（小四号宋体，1.35倍行距）

……

……

1

第2章 发动机气缸的磨损

（小二号黑体居中，上下各空一行）

2.1 活塞-缸套振动机理 （四号黑体）

2.1.1 测点布置 （小四号黑体）

□□实验在东北林业大学汽车质量监督检验站发动机实验室进行测试所用的发动机为EQ6100 型发动机，该机主要技术规格为六缸四冲程直列水冷式增压干式汽油机。缸径100mm,活塞行程115mm，最大功率不小于99千瓦。最大转速3000转/分，点火顺序为1—5—3—6—2—4；传感器的安装要考虑实用方便，测点位置的确定主要考虑对故障有一定的敏感性，测点位置如图所示， 为便于数据处理，在测量发机表面振动信号的同时，测取上止点信号和气缸压力信号。

（小四号宋体，1.35倍行距）

……

……

2.2 本章小结（四号黑体）

2

结□□论

（小二号黑体居中，不加章节号，上下各空一行）

□□（小四号宋体，1.35倍行距）

3

参考文献示例

参考文献

（小二号黑体，上下各空一行）

[1]余敏.出版集团研究[M].北京：中国书籍出版社，2001：179-193.

[2]昂温 G，昂温 P S.外国出版国史[M].陈生铮，译.北京：中国书籍出版社，1988.

[3]全国文献工作标准化技术委员会第七分委员会.GB/T 5795-1986 中国标准书号[S].北京：中国标准出版社，1986.

[4]辛希孟.信息技术与信息服务国际研讨会论文集：A集[C].北京：中国社会科学出版社，1994.

[5]孙玉文.汉语变调构词研究[D].北京：北京大学出版社，2000.

[6]顺炎武.昌平山水记：京东考古录[M].北京：北京古籍出版社，1982.

[7]王夫之.宋论[M].刻本.金陵：曾氏，1845（清同治四年）.

[8]赵耀东.新时代的工业工程师[M/OL].台北：天下文化出版社，1998[1998-09-26].http://www.ie.nthu.edu.tw/info/ie.newie.htm(Big5).

[9]PIGGOT T M.The cataloguer’s way through AACR2:from document receipt to document retrieval[M].London:The Library Association,1990.

[10]PEEBLES P Z,Jr.Probability,random variable,and random signal principles[M].4th ed.New York:McGraw Hill,2001.

[11]YUFIN S A. Geoecology and computers: proceedings of the Third International Conference on Advances of Computer Methods in Geotechnical and Geoenvironmental Engineering, Moscow, Russia, February 1-4,2000[C].Rotterdam: A. A. Balkema,2000.

……

……

（小四号宋体，1.35倍行距）

4

致谢示例

第1章 绪 论

## 1.1 研究背景

近年来，随着互联网产业的发展，大数据时代应运而生，现今互联网已经融入到大众生活的点点滴滴中，成为人们生活中不可或缺的一部分，人们可以通过各种网络设备：智能手机、平板电脑、个人PC等接入到互联网中，人们借助互联网实现：远程办公、移动支付、网络购物、在线交易、网络娱乐、了解新闻时事等，互联网包罗万象，将人类社会各个方方面面联结在一起组成一张巨大的信息网。我们正处在信息化时代，网络上的信息种类越来越多、数量越来越庞大。社交网络、电子商务、网络互娱、信息传媒等每时每刻都在生产大量信息，人类被各种数据带入了大数据新时代，在互联网时代，人们越来越依赖通过互联网来获取信息、娱乐消遣，互联网丰富了业余生活；与此同时，互联网也充斥着大量无用信息、甚至垃圾信息，随之而来的问题是：用户如何在有限的时间内高效地找到自己的所需要的信息，内容提供者如何展示自己所提供的信息给最需要的人，即内容提供者和内容消费者之间的匹配，这成为了互联网时代所面临的问题。

在使用视频网站的过程中，假如用户明确知道自己的刚需，这个时候用户可以借助搜索引擎来查找得到自己所需要的信息，例如用户想看《复仇者联盟3》，此时用户可以在视频网站中搜索这部电影进行观影，但是如果用户比较无聊，处在一个需求不明确的时期应该如何处理，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问题叫：信息过载（Information Overload）。

为了处理信息过载问题，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 研究意义

在实际使用中，我们发现视频网站总能贴用户心意地推荐合适的电影、电视节目，那么这些视频网站（内容提供者）是如何为用户（内容消费者）推荐喜欢的电影，这个过程的背后原理是什么，视频网站如何掌握用户喜好。首先，从信息获取的角度将人类获取信息的方法分为主动搜索与被动推荐的方式：首先以线下分析：一个人想要主动知道了解信息，首先会问周围认识的人是否知道，如果知道成功获取信息，如果不知道但知道可能知道就会得到一条线索，如果什么都不知道，那这条线索就断掉，将获取到的线索集合再次搜索；被动推荐是：你的朋友猜你想知道这个，问你我有一个消息你有没有兴趣知道来主动告诉你。

再类比到线上，主动搜索和被动推荐分别对应搜索引擎和推荐系统，搜索引擎是依靠用户主动在庞大的影视资源中搜索自己所心仪的电影、电视节目，但是主动查找需要很强的目的性，否则会浪费用户大量无效时间；而推荐系统虽然节省了用户时间，但是却将问题交付给内容提供者来处理，网站需要花大量时间获取用户数据、了解用户喜好，这个过程中充满各种问题。

个性化推荐系统通过用户数据用推荐算法来帮助用户进行查找和筛选信息，为用户提供充足、有效、合适、多样、新颖的信息帮助用户排忧解难，有效的推荐系统可以帮助减少用户的搜索时间，提高用户体验，但是推荐系统需要具有尽可能快的运行、响应速度、尽可能少的资源占用，既要提高效率，又要提高速度，因此对于推荐算法的优化成为解决这项问题的重要手段。

## 1.3 研究目的

通过建设一个小型的视频网站的过程来研究协同过滤推荐算法的原理和实现方法，利用推荐系统优化视频网站，赋予网站个性化的功能，能够为不同的用户定制不同的主页，推荐用户所喜欢的电影、电视节目。

在推荐算法系统中进行拓展：当使用不同的相似性算法时来验证出最适用的推荐算法，并尝试通过优化推荐算法或者提高计算能力来优化推荐系统的效率，一个优秀的推荐系统需要保证实时性、高效性、具有最高的利用率的系统才能具有商业价值。

研究推荐算法的评价性算法，在推荐系统的研究过程中，评价推荐结果的标准是一个非常基础且重要的前提，只有尽可能准确的评价才有推荐算法优化的前提。

## 1.4 本文安排

第一章，绪论。首先介绍本课题研究背景、研究意义和目的。

第二章，研究现状分析，确认当前协同过滤的研究现状，确定从何角度进行研究。

第三章，推荐算法及其相关介绍。（1）介绍常用的推荐算法以及相关概念，对推荐算法流程，原理进行简要概述；（2）对于推荐算法依赖的核心相似性算法进行简要概括，实验不同相似性算法对于实验的影响；（3）对于推荐算法的评价指标进行研究，如何评价一个推荐系统的各项参数。

第四章，协同过滤推荐算法的具体应用，确认

第五章，基于协同过滤电影推荐系统的设计与实现

第六章，总结与展望，优化与修改

第七章，总结与展望

第2章 研究现状及分析

## 2.1 推荐算法的发展

推荐算法和搜索引擎相似，但是搜索是主动的，而推荐系统根据用户的历史记录进行用户喜好推荐，用户处于一个被动的状态，推荐系统往往作为一个应用网站的子系统来进行设计，依赖网站的数据来进行推荐活动，而非单独存在。在互联网的各类网站中，推荐系统应用很广，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。

个性化推荐算法的研究开始于上世纪90年代，随着互联网的发展开始应用于各行各业，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。个性化推荐系统按照内容可以分为（1）用户模块，即推荐的目标人群；（2）内容模块，即待推荐的主要内容；（3）推荐算法，将内容推荐给用户的规则，是连接用户与内容的核心桥梁，也是需要深入研究的内容。

## 2.2 推荐算法的研究现状

当前主流的推荐系统主要分为：基于内容的推荐系统（Content Based, CB）、协同过滤推荐系统（Collaborative Filtering,CF）、混合推荐系统，其中混合推荐系统是一种综合性推荐系统，是融汇各种推荐算法优点来设计，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。

协同过滤推荐算法（Collaborative Filtering,CF）目前应用较多，是一种成熟的推荐算法，目前在工业生产中被广泛推荐。其基本思想是根据用户历史行为数据推荐用户喜好。CF算法主要有三个流程：（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，根据历史数据确定相似性矩阵，对用户或者内容进行相似性计算；（3）相似性推荐，根据相似性矩阵为用户进行推荐。根据推荐系统要处理的问题可以分为：（1）评分预测；（2）Top N 推荐。

在推荐系统的发展中长期存在着一些问题：冷启动问题；数据稀疏问题；用户喜好偏移问题。冷启动问题即新用户、新项目没有历史数据的情况下如何加入到推荐系统中来；数据稀疏的问题是指海量的用户和项的情况下，用户数据稀少，导致用户相似性矩阵或物品相似性矩阵稀疏；用户兴趣漂移是指用户的兴趣变化，因为人是具有思维的个人，具有相当大的不确定性、偶然性有可能某一天用户的兴趣突然就变化180度，那么对于原先的相似性矩阵就无法适用了。

## 2.3 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、不可遍历，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，腾讯视频网站可以根据电影之间的相似性进行归类，根据用户播放记录帮助用户筛选用户可能感兴趣的电影；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，来推荐用户好友圈子中受欢迎的电影节目。

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来计算每首音乐之间的相似性来，通过用户的听歌记录来生产用户的私人歌单，通过所有用户的播放记录找到两个听歌风格相似的用户来互相推荐歌曲。

阅文网站的小说推荐，当用户浏览一本小说时，下面会给出推荐列表，从小说的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，现在电商平台中的推荐系统就成了标配，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，掌握每一个人的性格、喜好等，通过用户喜好进行用户推荐。

在推荐列表应用中包含着商品的显性信息和隐形信息以及相关属性，其他用户对商品的评分，综合评价出两个商品的相关性。显性的信息是两个商品明面上的相似处，例如两本书之间的相似，而隐形的相似性是从用户的购买记录中发现的相似规律，例如笔记本和笔会经常被用户同时购买。在亚马逊的个性化推荐系统中，最重要的算法是一种基于物品的推荐算法，给用户推荐用户喜欢物品的相似物品，另一种推荐算法是基于用户的推荐算法，亚马逊是利用Facebook上的好友信息，判定好友之间的相似度来推荐商品。使用推荐系统的直接好处是给亚马逊公司带来直接的经济效益，据统计，亚马逊百分之三十左右的商品销售来自于推荐系统。

第3章 推荐算法以及相关介绍

## 3.1推荐算法总览

### 3.1.1基于内容的推荐（Content Based，CB）

主要用于推荐文本类项目，采用浅层模型分析用户历史记录，提取核心内容词汇，根据物品的便签属性、内容关键词等用户特有的兴趣相关的词汇，推荐有相似标签的物品（文本、商品、电影等）。例如：一个人喜欢看《夏洛特烦恼》那么我可以给他推荐同为喜剧的《人在囧途》、《飞驰人生》等，他们的内容标签可以为：类型喜剧，主演：沈腾、尹正，例如一个人购买了数据结构书籍，可以根据内容推荐其相关算法书籍。不能根据评分或名称，需要根据物品本质进行推荐。 比较最直观，推荐理由也很充分，但需要做好新物品的归类，内容标签的标定，并且推荐精度较差，结果惊喜度不足，你所推荐的正式用户所想的。

### 3.1.2 基于内存的协同过滤（Memory-Based Collaborative Filtering）

协同过滤(Collaborative Filtering, CF)是当前推荐算法中比较常用和经典的推荐算法，种类繁多，目前在生产环节又很多的应用，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、物品或者内容间的相关性进行推荐，如果用户喜欢A，那么就像和B相似度B推荐给用户，协同过滤可细分为基于内存的协同过滤、基于模型的协同过滤，其中基于内存的协同过滤可以细分为基于用户的协同过滤推荐（User Based CF）、基于项目的协同过滤推荐（Item Based CF）：

基于用户的协同过滤（User-based Collaborative Filtering）

User CF基本思想是根据用户行为记录发现其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来为用户推荐，如图，小张和小明观影记录相似可以得出其有共同喜好，根据小张的行为为小明推荐电影3。User CF更接近于社会化推荐，比较精准，适用物品多且杂的场景，时效性较强的场合，具有推荐新信息的能力，可以发现用户潜在但自己尚未察觉的潜在兴趣,惊喜度高，能够推荐艺术品、音乐、电影等难以进行内容分析的产品，具有跨领域能力。代价是运算量很大，具有冷启动的问题（对于新来的人历史记录少，获取的信息不足少)，当用户间的共同行为较少，难以形成有意义的邻居集合时，难以进行有效的推荐，并且用户间距离可能变化很快，因此不适合线上实时计算。

传统的User CF实现可以分为以下过程：

根据用户与系统的交互行为，收集用户的行为数据如表（用户行为数据表），建立用户-项评分矩阵，用户的历史行为数据多样包括：用户浏览、用户观看、用户打分、用户频率（搜索次数、观看次数、点击次数）、用户点赞分享收藏等行为，本论文主要研究用户的主动打分数据进行分析，首先建立用户评分矩阵如图：

|  |  |  |
| --- | --- | --- |
| **userId** | **movieId** | **rating** |
| 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 |

用户行为数据表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **userId** | **movieId** | | | | | | |
|  | 1 | 3 | 6 | 5 | 7 | 10 | 2 |
| 1 | 4 | 4 | 4 |  |  |  |  |
| 2 |  | 5 |  | 5 | 3 | 5 |  |
| 3 | 4 | 5 |  | 5 |  |  |  |
| 4 |  | 5 |  | 3 |  |  | 5 |

用户-电影矩阵

根据上述用户-电影评分矩阵中的用户对电影的评分数据，在结合相似性计算方式，确定偏好相似的用户集，不同的相似度计算方法最终会得到不同的结果。

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

基于项目（Item-based Collaborative Filtering）

基于项的协同过滤,主要考虑项与项之间的联系，通过确定相似的项目集合，在根据用户在系统的行为信息进行推荐。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、各种猜你想听、想看等，同时Item CF还可以给推荐提可信的推荐理由，例如：豆瓣的「喜欢物品A的人也喜欢物品B」和亚马逊的「买了物品A的人也买了物品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、更新快的场景则效果不足，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。

Item CF实现可以分为以下过程：

（1）首先根据用户的历史行为数据，如表（用户行为数据表），建立用户-项评分矩阵，用户的历史行为数据多样包括：用户浏览、用户观看、用户打分、用户频率（搜索次数、观看次数、点击次数）、用户点赞分享收藏等行为（同User CF），本论文主要研究用户的主动打分数据进行分析，首先建立用户评分矩阵如图：

|  |  |  |
| --- | --- | --- |
| **userId** | **movieId** | **rating** |
| 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 |

用户行为数据表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **userId** | **movieId** | | | | | | |
|  | 1 | 3 | 6 | 5 | 7 | 10 | 2 |
| 1 | 4 | 4 | 4 |  |  |  |  |
| 2 |  | 5 |  | 5 | 3 | 5 |  |
| 3 | 4 | 5 |  | 5 |  |  |  |
| 4 |  | 5 |  | 3 |  |  | 5 |

用户-电影矩阵

（2）根据上述用户-电影评分矩阵中的用户对电影的评分数据，在结合相似性计算方式，物品-物品相似矩阵，并且不同的相似度计算方法最终会得到不同的结果。

（3）根据物品相似度矩阵，结合用户电影矩阵，找到和用户已观看电影相似的电影列表，生成推荐电影Top N列表。

### 3.1.3 User CF 和 Item CF 的对比

User CF 是推荐系统的领域中比较久远的算法，同样是基于内存的协同过滤算法User CF 和 Item CF 拥有着相似的流程。对比异同：

计算主体。User CF 主要计算用户（User）之间的相似性，最终得到用户-用户相似性矩阵（User-User-Similarity）；而Item CF 主要计算项目（Item）之间的相似性，最终得到物品-物品相似性矩阵（Item-Item-Similarity）所以User CF 适合用户比较少的场景，Item CF 适合项目的数量比较少的场景，否则计算量会随着数量级增加而大大增加。

应用领域。User CF 倾向于找到用户临近的聚类，为用户找到共同爱好群，相当于是一个同好交流群，并且对于新项目响应比较慢，不适合个性化强的场景；Item CF 则是为项目找到聚类，对于用户的新项目的响应较为敏感，个性化比较明显。

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，只适合离线推荐；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，如果用户行为发生变化，结果一定发生变化。

如何User CF 相似度每隔一段时间需要重新离线计算一次，所以新物品上线一段时间后才能被加入推荐列表，Item CF 在新物品加入时可以进行在线计算，立即加入推荐列表中。

|  |  |  |
| --- | --- | --- |
|  | User CF | Item CF |
| 性能 | 运算量与用户数量有关 | 运算量与物品数量有关 |
| 领域 | 时效性高，个性化低 | 物品丰富，个性化要求 |
| 实时性 | 不足 | 用户行为变化响应较快 |
| 冷启动 | 新用户不友好 | 新用户、新物品可以快速加入推荐 |
| 推荐理由 | 按照相似用户群体推荐 | 根据用户看过推荐 |
| 跨领域性 | 跨领域推荐 | 同类物品推荐 |
| 核心矩阵 | 用户相似性矩阵 | 物品相似性矩阵 |

### 3.1.4 基于模型的协同过滤（Model-based Collaborative Filtering）

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。

基于模型的协同过滤算法，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。常规的应用在推荐中的机器学习方法包括关联分析、聚类算法、回归算法、分类算法、矩阵分解、神经网络、隐语义模型、图模型等，随着神经网络的研究和发展，基于神经网络的推荐算法也日渐壮大。适用范围广大，可以应对数据稀疏的场景，能很好地发现用户潜在偏好。

### 3.1.5 混合推荐算法

在上文的算法都是单一的推荐算法，每个算法都有各自的有点和缺点；混合推荐算法不是一种算法的名称，而是多种推荐算法的混合使用得到的一种新的推荐算法，集百家之长，虽然提高了算法的复杂度，但是可以利用各种算法的优缺点进行互补。

## 3.2 相似度算法总览

推荐算法中具有一个核心的问题：相似度的判断，如何说明两个项之间的相似性，需要将具体的物体转化为一个数据向量通过向量之间的相似性来比较。因此需要采用合适的相似性算法来进行。

欧几里得度量（Eucledian Distance）欧氏距离是最常用的距离计算公式，即连接两个点的线段的长度，衡量多维空间中两个点之间的绝对距离，计算简单，使用勾股定理在笛卡尔坐标计算距离，当数据很稠密并且连续时，这是一种很好的计算方式。缺点：欧式距离受到尺度影响，这意味着所计算的距离会根据特征维度的单位不同发生倾斜，比如对身高（cm）和体重（kg）两个单位不同的指标使用欧式距离会使结果不准确。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，欧式距离在不同尺度具有风险。

曼哈顿距离（Manhattan Distance）也叫出租车距离，正正方方街区的城市中，出租车从A点到B点的距离。

切比雪夫距离 （Chebyshev Distance）

闵可夫斯基距离 （Minkowski distance）是一组距离的定义，是对多个距离度量公式的概括性表述：

例如当p等于无穷大时,作为切比雪夫距离

当p=2时，作为欧式距离

当p=1时，作为曼哈顿距离

余弦相似度 （Cosine Similarity）,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度，将向量分布到直角坐标系（n维向量分布到n维空间直角坐标系中），绘制从零点出发的向量空间，求两个向量之间夹角余弦值，表示这两个向量的相似性，当夹角越小，余弦值越接近于1，它们的方向越吻合，则越相似，反之则不相似。

杰卡德相似度 （Jaccard Similarity）对于两个集合A、B，利用A、B交集元素的个数在A、B并集中所占的比例来确定集合的相似度，称为这两个集合的杰卡德系数，用符号 J(A,B)表示，是衡量两个集合相似度的一种指标。

## 3.3 推荐系统评测

### 3.3.1 评测方法概述

如何评价一个推荐系统的优劣，怎样的推荐系统是我们所需要的，目前还没有一个统一的标准，但是推荐系统终将被应用到实践之中，因此用户的满意度是最佳的评价，但客户满意度需要投入到应用中去，消耗的人力、物力大，并且具有很大风险，况且用户满意度作为主观判断无法用数值衡量。常规的推荐系统评价流程为：离线实验（offline experiment）、用户调查（user study）和在线实验（online experiment），测试的范围从小到大，在现有的推荐算法中，有些算法能够利用离线指标判断，但是离线指标并不能准确评测算法质量，在电影推荐中，用户的评分具有非常强大的主观性：用户会因为电影质量评价高分，也有可能因为某些原因给以虚假的评分；就算数据准确，计算到的推荐结果也无法完全的与用户实际表现对应，对于独立思考的个性的人来说，具有很大的变化性。

**1）离线实验**

离线实验通常采用数据集进行测试、常用于开发阶段的测试，具体过程如下：

a）获得用户行为数据，例如用户的购买记录、播放记录、评分记录、收藏记录等，需要事先收集测试数据集合的过程；

b）将数据集按照一定的规律分成训练集和测试集；

c）在训练集上训练推荐模型并利用测试集进行预测；

d）利用事先定义的离线指标评测算法在测试集上的预测结果。

e）对于推荐算法进行改进

离线实验的优点是：

不需要用户参与成本低

利用离线数据验证、时间短、效率高

可以无人值守，时间地点方式灵活

离线实验的缺点是：

受到数据集合的局限性、适用范围的影响，例如一个数据集中没有包含某用户的历史行为，则无法评价对该用户的推荐结果，一旦数据集不准确就无法为用户准确预测；对于评价结果缺乏主观性，无法考虑到人的主观思想变化，缺乏灵活性；难以找到离线评价指标和在线真实反馈(如搜索率、点击率、转化率、点击次数、时长、购买客单价、购买商 品类别等)之间的关联关系；对于用户的变化无法做到及时响应，例如人的兴趣突然偏移等

**2）用户调查**

用户调查需要真实的用户在待测推荐系统上完成任务，并且观察和记录用户的行为以及回答一些问题，分析用户的行为和答案，了解测试系统的性能。

用户调查的优点是：

可以获得用户主观感受的指标，对于用户的主管想法进行记录和考究；测评范围小，造成风险的影响范围较小，风险易控制。

用户调查的缺点是：

对于测试用户招募消耗巨大；无法组织完全覆盖的测试用户，统计意义不足。

**3）在线实验**

通过离线实验和用户调查测评之后，可以将系统上线做AB测试，即将新的推荐系统与旧的推荐系统共同投入使用进行测试，通过两个推荐系统的对比进行测评，并且减少风险。

在线实验通过一定的规则将用户随机分成几组，对不同组的用户采用不同的推荐算法，然后通过统计不同组的评测指标，比较不同算法之间的优劣，其的核心思想是:

a) 控制变量多方案共同运行

b) 比较各个方案测试结果

c) 以某种规则进行修改优化或者是淘汰

在推荐系统的评价中，唯一变量就是推荐算法的选择。

在线实验的优点是：评价结果通过在线考验，具有很强的参考性，能比较准确的反应推荐系统性能。

在线实验的缺点时：评价过程比较复杂，对于时间、人力、物力耗资巨大；具有不可控的风险，有造成潜在经济效益损失的风险。

3.3.2评测算法的选择

离线实验评估算法

由于条件的限制，本次采用离线测评，即根据推荐系统在离线数据集合中的表现，基于一些离线评估指标来衡量推荐系统的效果，相比较在线系统，离线系统更加经济、方便，利用验证数据集来测试推荐系统的效果。本论文采用的评估环节有两个：拆分数据集、选择离线评估指标。

a）数据集拆分。在机器学习过程中，通常需要将数据集拆分为训练数据集、验证数据集，测试数据集合。它们的功能分别如下：

训练数据集（Train Dataset）：用来进行机器学习模型的构造。

验证数据集（Validation Dataset）：辅助模型构造，评估模型。

测试数据集（Test Dataset）：评估训练完成的模型各项指标。

本论文推荐系统采用协同过滤推荐算法，暂不涉及机器学习方向，因此将数据集合随机分为训练数据集合（Train DataSet）、测试数据集合（Test DataSet）。

b）离线评测指标的选择。在分类模型评判的指标中，常见有1.混淆矩阵以及其指标 2. ROC曲线 3. AUC面积进行评判。

混淆矩阵（也称误差矩阵，Confusion Matrix）：混淆矩阵是评判模型结果的指标，属于评估模型的一种，多用于判断分类器（Classifier）的优劣，在预测分类实验中将真实结果、预测结果真假分布到2 x 2的表格中进行分析，在表格里可以清晰的看到每个类别识别正确和识别错误的数量。

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

混淆矩阵的指标：

TP（True Positive）：将真值预测为真，真正类，正确预测

TN（True Negative）: 将真值预测为假，伪假类，错误预测

FP（False Positive）: 将假值预测为真，伪正类，错误预测

FN（False Negative）：将假值预测为假，真假类，正确预测

根据对基础指标的归类得到了二级指标进行结果的衡量：

准确率（Accuracy）：预测正确的数量（真正类+真假类）与所有预测数量之比

精确率（Precision）：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

召回率（Recall）：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

AUC （ area under the curve ）。ROC曲线向下、向右与x、y轴的面积

ROC（receiver operating characteristic curve），利用FPR做横坐标，TPR做纵坐标，绘制曲线，其中FPR为假正例率，TPR真正例率，如下：

FPR：真正类的数量在所有预测为正类的数量的比例

TPR：伪正类的数量在所有预测为假类的数量的比例

F分数（F score）：精确率(Precision)和召回率(Recall)评估指标,理想情况下希望精确率和召回率都高，但一般情况下，精确率高，召回率就低，召回率高，精确率就低，所以在实际中需要做出取舍，例如：在保证召回率的条件下，尽量提升精确率。在癌症检测、地震检测、金融欺诈等情况下，需要先保证精确率再尽量提升召回率。

F-score，综合Precision和Recall的调和指标：

当β=1时，称为 F1 score，这时，精确率和召回率都很重要，权重相同。当精确率更重要时，那就调整β的值小于1，当召回率更重要时，那就调整β的值大于1。

F1 Score：

评分预测评价指标

均方误差MAE（Mean square error）

均方根误差RMAE （Root Mean Square Error）

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

本论文实验使用Windows系统环境配置，具体信息为：Windows 10 家庭版64位操作系统、处理器为Intel (R)Core(TM)i5-7300HQ、内存16 G，可用磁盘300 G固态。编程语言为Python，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、能偶跨平台使用，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，具体环境为：Python 3.6 、VS Code。

实验采用Movies Lens 数据集作为本次实验的训练、测试数据，该数据集由GroupLens实验室采集上传的电影评分数据信息表，由于机器性能受限，采用ml-latest-small数据集，包括600多个用户9000部电影，1万条评分信息。

### 4.1.1 Item CF 实验设计

实验1：基于杰卡德相似度算法的Item CF实现可以分为以下过程：

1. 首先读入用户的历史行为数据表（用户行为数据表），
2. 建立如下所样式的movie-user矩阵

|  |  |
| --- | --- |
| movie\_user = {}  for user, movies in trainSet.items():  for movie in movies:  if movie not in movie\_user:  movie\_user[movie] = set()  movie\_user[movie].add(user) | |
| movie | set() |
| movie1 | user1,user2,user3,user4 |
| movie2 | user2,use3,user4,user5,user6 |
| movie3 | user5,user6,user9 |
| movie4 | user3,user6,user7 |

1. 利用电影之间的观影数据的交集比并集算出电影的相似性

核心程序：

# 计算交集比

def jacard(u1,u2):

return len(set.intersection(u1, u2)) / len(set.union(u1,u2))

1. 根据物品相似度矩阵，结合用户电影矩阵，找到和用户已观看电影相似的电影列表，生成推荐电影Top N列表。

|  |
| --- |
| def recommend(user, K, N, trainSet, movie\_sim\_movie):  rank = {}  watched\_movies = trainSet[user]  for movie, rating in watched\_movies.items():  # 遍历与物品item最相似的前k个产品，获得这些物品及相似度  for related\_movie, w in sorted(movie\_sim\_movie[movie].items(), key=itemgetter(1), reverse=True)[:K]:  if related\_movie in watched\_movies:  continue  # 计算用户user对related\_movie的偏好值  rank.setdefault(related\_movie, 0)  # 通过电影评分\*电影相似度计算推荐的电影可能评分，按照预测评分返回推荐top N  rank[related\_movie] = w \* float(rating)  return sorted(rank.items(), key=itemgetter(1), reverse=True)[:N] |

由于杰卡德相似度无法体现用户评分数据，因此利用相似度和用户观影的电影评分估计出推荐电影评分。

实验2：基于余弦相似度算法的Item CF实现可以分为以下过程：（仅描述不同步骤)

（1）建立movie-user矩阵时，需要对空余值补0，这里使用numpy生成0数组

|  |  |
| --- | --- |
| # 将不足n维的movie向量转为n维向量，余弦相似度需要保证两向量同维度  def to\_array(dic,l=610):  n\_x = np.zeros(l)  for i,v in dic.items():  n\_x[i-1] = v  return n\_x | |
| movie | List() |
| 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 |

（2）使用余弦相似度公式计算电影之间相似度

核心程序：

|  |
| --- |
| # 相似性算法  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) |

实验3：尝试结合余弦相似度和杰卡德相似度算法，利用杰卡德相似性算法和交集的余弦相似性算法结合

其中A1、B1为A、B具有交集的项

核心相似度算法：

|  |
| --- |
| def cosine\_jacard(movie1,movie2):  # jacard 相似度  index = list(set.intersection(set(movie1), set(movie2)))  un = list(set.union(set(movie1), set(movie2)))  j = len(index) / len(un)    """交集的余弦相似度"""  x = to\_array(movie1)[index]  y = to\_array(movie2)[index]  xy = np.sum(x\*y)  x = np.linalg.norm(x,2)  y = np.linalg.norm(y,2)  return xy \* j / (x\*y) |

### 4.1.2 User CF 实验设计

实验4：基于杰卡德相似度算法的User CF

相似度计算过程：

（1）首先读入用户的历史行为数据表（用户行为数据表）

（2）建立user\_movie矩阵，形如：

|  |  |
| --- | --- |
| user | set() |
| user1 | movie1,movie2 ······ |

核心代码

|  |
| --- |
| def calc\_user\_sim(trainSet):  # 构建user\_movie矩阵  user\_movie = {}  for user, movies in trainSet.items():  for movie in movies:  if user not in user\_movie:  user\_movie[user] = set()  user\_movie[user].add(movie)    # 建立user\_sim\_matrix矩阵 用户间相似性矩阵  user\_sim\_matrix = {}  print('Build user co-rated movies matrix ...')  for u1, m1 in user\_movie.items():  for u2, m2 in user\_movie.items():  if u1 == u2:  continue  else:  user\_sim\_matrix.setdefault(u1, {})  user\_sim\_matrix[u1].setdefault(u2, 0)  user\_sim\_matrix[u1][u2] = len(set.intersection(m1, m2)) / len(set.union(m1,m2))  return user\_sim\_matrix, user\_movie, movie\_count |

（3）根据上述用户间相似性矩阵中的用户对电影的评分数据，在结合相似性计算待推荐电影的评分数据，生成推荐电影Top N列表。

推荐代码

# 针对目标用户U，找到K部相似的电影，并推荐其N部电影，

def recommend(user, K, N, trainSet, user\_sim\_matrix):

rank = {}

watched\_movies = trainSet[user]

# v=similar user, wuv=similar factor

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)

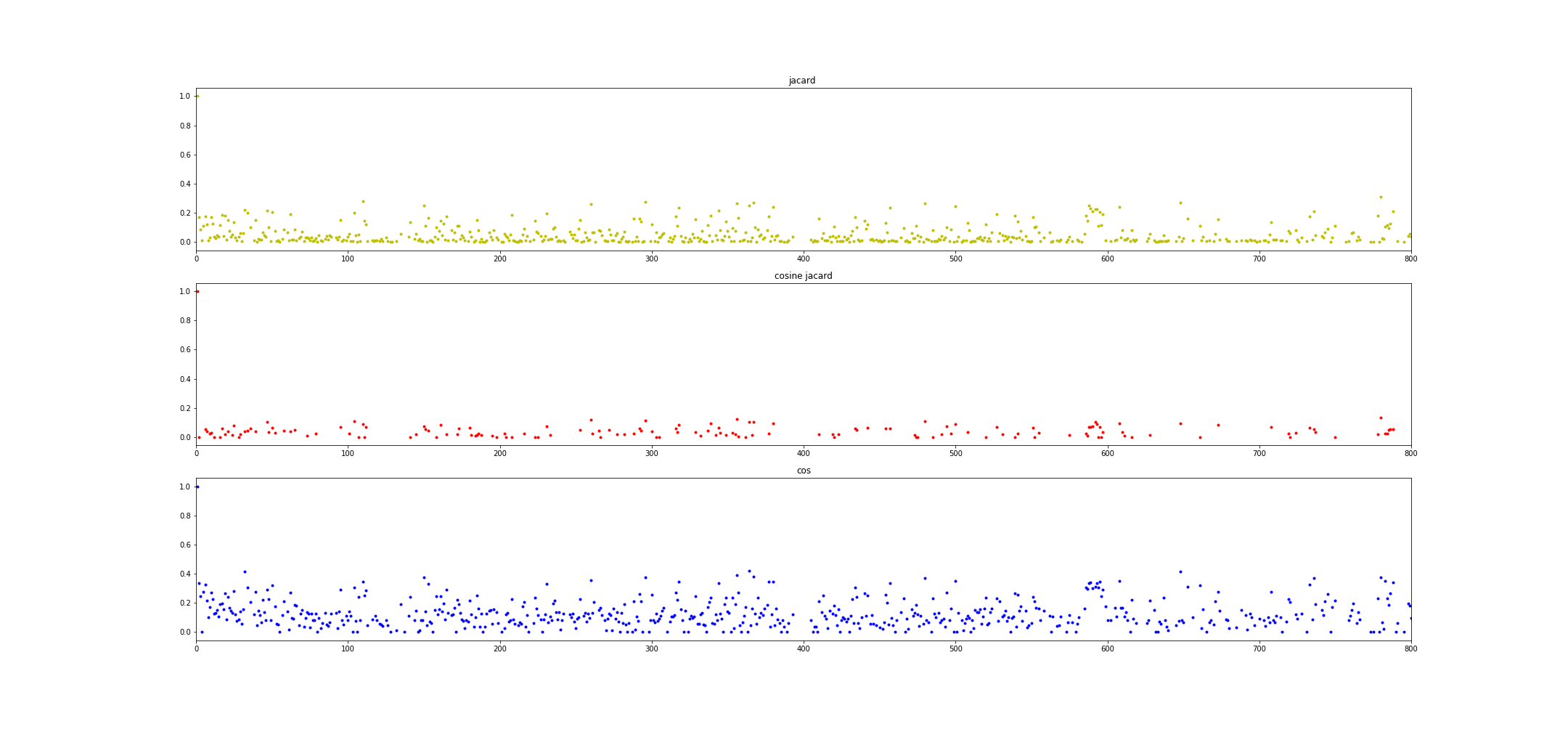
rank[movie] = wuv \* rating

return sorted(rank.items(), key=itemgetter(1), reverse=True)[0:N]

## 

## 4.2 协同过滤推荐算法分析

相似性算法结果集合：截取800部电影。



由图可以观察：三种相似度算法的峰谷接近，相似度趋势相似，在余弦相似度的结果中，相似度结果之间区分度比较高，结果散步在0到0.4之间，在杰卡德的算法中，结果散布0到0.2之间。

三种相似度算法计算相似性的运行时间：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 杰卡德相似度 | 余弦相似度 | 杰卡德结合交集余弦相似 |
| 开始时间 | 21:12:40.193205 | 12:47:20.921754 | 13:52:50.977521 |
| 结束时间 | 21:15:00.385559 | 13:14:54.359994 | 14:35:40.747132 |
| 用时 | 00:02:20.192354 | 00:27:33.438240 | 00:42:49.769611 |

基于用户的协同过滤算法运行时间

|  |  |
| --- | --- |
| 开始时间 | 08:24:47.405325 |
| 结束时间 | 08:24:51.146901 |
| 用时 | 00:00:03.741576 |

杰卡德相似算法，基于项的推荐结果

|  |  |
| --- | --- |
| [(166, 3.333333333333333), (2607, 3.333333333333333),  (2769, 3.333333333333333), (3078, 3.333333333333333),  (3747, 3.333333333333333), (3862, 3.333333333333333),  (5347, 3.333333333333333), (5423, 3.333333333333333),  (5425, 3.333333333333333), (5486, 3.333333333333333),  (6335, 3.333333333333333), (7354, 3.333333333333333),  (7386, 2.5), (118, 2.5),  (521, 2.5), (835, 2.5),  (1040, 2.5), (2095, 2.5),  (2425, 2.5), (2552, 2.5)] | |
| 开始时间 | 08:43:37.653993 |
| 结束时间 | 08:43:38.036356 |
| 用时 | 00:00:00.382363 |

杰卡德相似算法，基于项的推荐结果

|  |  |
| --- | --- |
| [(1089, 78.82580696217214), (1196, 61.282928846791435),  (1210, 57.77731956398587), (1527, 57.641792240098006),  (2916, 50.21300416057626), (1240, 49.40658980557108),  (2918, 48.09092229144081), (2683, 41.613716850034905),  (296, 41.16519846085778), (1682, 40.73998296885274),  (2858, 39.9216220944197), (858, 39.22801247072661),  (2997, 36.45176676147006), (589, 36.39426872679257),  (1097, 36.03207582139306), (1036, 34.97743872861351),  (3033, 34.08231204279323), (1580, 31.63422636389324),  (1129, 30.260999081947112), (2959, 28.552920447178778)] | |
| 开始时间 | 17:14:41.179340 |
| 结束时间 | 17:14:41.766632 |
| 用时 | 00:00:00.587292 |

杰卡德相似算法，基于用户的推荐结果

|  |  |
| --- | --- |
| [(199, 0.13819095477386933), (242, 0.13819095477386933),  (627, 0.13819095477386933), (695, 0.13819095477386933),  (896, 0.13819095477386933), (1175, 0.13819095477386933),  (1619, 0.13819095477386933), (1900, 0.13819095477386933),  (1916, 0.13819095477386933), (2275, 0.13819095477386933),  (2333, 0.13819095477386933), (2568, 0.13819095477386933),  (2574, 0.13819095477386933), (3266, 0.13819095477386933),  (3484, 0.13819095477386933), (3909, 0.13819095477386933),  (4278, 0.13819095477386933), (4298, 0.13819095477386933),  (4562, 0.13819095477386933), (4901, 0.13819095477386933)] | |
| 开始时间 | 09:12:24.804784 |
| 结束时间 | 09:12:24.806811 |
| 用时 | 00:00:00.002027 |

## 4.3 基于协同过滤推荐算法的推荐系统评测

第5章 电影推荐系统的实现

## 5.1 技术基础

Flask Flask是目前比较流行的基于Python的Web框架，是用Python来实现的。其显著的特点是：“微”框架。表示Flask是一个核心的简单、方便入手、同时又易于扩展的轻量级框架。默认的Flask是不包含诸如数据库支持、表单验证，或者是其他任何功能。但是易于拓展的Flask框架支持使用扩展方式来给应用添加其他功能。这些扩展包括数据库集成、表单验证、文件上传处理、其他各种各样的开放认证技术等功能，开发者可以根据自己的需求选择自己的功能，因此Flask在Web开发方面变得非常流行,本论文中推荐算法采用Python实现，因此采用作为Flask开发视频网站的技术支持。

MongoDB是一个基于分布式文件存储的文档型数据库，介于关系型和非关系型之间的模式，是NoSql数据库中最热门的一种，基于C++语言开发的高性能、易部署，作为NoSql型数据库，旨在为大规模web应用部署服务，MongoDB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），存储高效，文件以BSON格式存储（Json格式的拓展）和Python有较好的适配，适合作为本论文中网站的后台数据库。

Redis 是一个开源的高性能 Key Value 数据库，Redis将数据存储在内存中，所以读写速度非常快，并且支持数据持久化，因此Redis被广泛应用在缓存方向，可以作为分布式锁来使用，本论文中，利用Redis作为缓存数据库。

视图展示功能使用，JinJa2 和bootstrap作为网页技术支持。Bootstrap是流行的HTML、CSS、JS框架，主要用于开发响应式布局，由Twitter的工程师为提高内部效率开发的易用、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。

NumPy(Numerical Python) 是基于Python 语言开发的一个开源数值计算扩展程序库，支持大量的维度数组与矩阵运算，也针对数组运算提供大量的数学函数库支持，运行速度非常快，本论文中利用NumPy提高推荐算法的计算速度。

## 5.2 数据库方面

数据库设计，核心数据表：

Ratings（用户观影记录表）:

|  |  |
| --- | --- |
| userId | 用户ID |
| movieId | 用户观看的电影 |
| ratings | 用户对该电影的评分 |

Movies（电影信息表）:

|  |  |
| --- | --- |
| movieId | 电影ID |
| title | 电影标题 |
| genres | 电影体裁、类型 |

Links（用户观影记录表）:

|  |  |
| --- | --- |
| movieId | 电影ID |
| imdbId | Top Story 网站的编号 |
| tmdbId | TMDB 网站的电影编号 |

具体链接拼接方式：

Top Story 网站：<http://www.imdb.com/title/tt+> imdbId

TMDB 网站的地址：link<https://www.themoviedb.org/movie/+ tmdbId

User\_rec\_cache（用户推荐电影列表缓存）

|  |  |
| --- | --- |
| UserId | 电影ID |
| movieList | Top Story 网站的编号 |

Movie\_sim\_cache（电影相似性矩阵缓存）

|  |  |
| --- | --- |
| UserId | 电影ID |
| movieList | Top Story 网站的编号 |

User\_sim\_cache（用户相似性矩阵缓存）

|  |  |
| --- | --- |
| UserId | 电影ID |
| movieList | Top Story 网站的编号 |

## 5.3 总体设计环节

在推荐算法的基础上、实现个性化电影网站的开发，将协同过滤推荐算法应用到电影网站之中，网站的主体为了展现推荐算法，故而比较简洁，电影数据取自MovieLens，内容包括电影名称、电影流派、跳转地址，网站首页根据用户的登录的账号不同，为每个用户展现不同的界面并且添加跳转链接。

1.视频网站首页

个人用户页面：用户个人的信息管理页面，负责用户个人信息管理

后台管理系统页面：后台管理系统是管理员维护网站功能的工具

网站首页：网站首页会根据用户来展示不同的推荐电影

视频播放页面：视频播放页面，播放页面具有电影详细信息、评分、评论区

搜索结果页面：展示用户搜索电影的后的结果

2.用户个人信息管理

用户登录、用户注册：由于HTTP协议是没有状态的，而网站需要记录用户登录状态，所以采用Session 和Cookies 来进行记录用户的登录信息，为防止Cookies欺骗、采用Token令牌来进行验证。

本论文采用MD5加密来生成Token：

代码中使用Ptyhon中的hashlib模块，利用用户名、密码、以及加盐固定字符串生成用户token

代码部分：

def get\_md5(username, password):  
 m = hashlib.md5()  
 m.update(bytes(username + password + 'the salt', encoding='utf-8'))  
 return m.hexdigest()

@app.route('/login', methods=['GET', 'POST'])

修改信息：用户修改个人信息

播放记录管理：用户修改个人的播放记录

3.用户行为管理

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

4.站内搜索功能

条件搜索：限制用户的查询条件的限制性搜索，仅可以搜索电影名、演员等

全局搜索：解除了用户查询条件限制，用户可以通过电影名、导演、演员、上映时间等进行精确或者模糊查询。全局搜索需要对用户的输入进行合法验证分析，需要使用全文搜索技术对数据库所有字段进行搜索。

5.全网搜索：对于站内不存在的资源需要调用全网搜索，可以使用百度搜索引擎接口，但是需要对搜索结果进行筛选、归类

6.推荐功能

用户喜好电影推荐：用户进入网站首页时，系统会给出用户推荐电影结果

7.数据分析功能：分析用户的播放记录

6.管理员功能（后台管理功能）

电影信息管理：电影信息的管理

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、播放时长、搜索次数、收藏人数等信息进行统计、并且依照这些信息计算出电影的热度

用户信息管理：管理员对网站平台用户的信息进行管理

用户行为查询管理：管理员对用户的操作记录（用户播放记录进行查询）

5.4 推荐算法模块：

网站设计中采用的算法为基于用户的协同过滤推荐算法和基于项的协同过滤推荐算法结合，将两种算法的推荐结果结合在一起为用户推荐电影。为了减少给用户推荐电影清单的运行时间，采用定期离线计算、利用MongoDB做推荐电影列表缓存，即当用户访问网站时，先从缓存中加载用户喜好，缓存的喜好通过设定时间进行更新，更新时间有两种：1. 用户下线时，根据用户新增的历史记录来离线计算用户喜好；2.用户在线播放电影时，计算出该电影的相似电影用以推荐。其中第一种方式是全面计算并更新用户的推荐电影列表，第二种是根据用户在线播放行为，为用户推荐用户在看电影的相似电影。

5.4 启动数据

MovieLens数据集是 GroupLens团队收集MovieLens网站的数据，其中包含多个用户对多部电影的评分数据，以及一些用户、电影相关信息，该数据集常用做推荐系统、机器学习算法的测试数据集。在推荐系统领域，该数据集被广泛应用，产出了很多的论文、文献。

下载地址为：[http://files.grouplens.org/datasets/movielens/](https://links.jianshu.com/go?to=http://files.grouplens.org/datasets/movielens/" \t "_blank)，有好几种版本，对应不同数据量，本论文采用的数据为movielens-latest-small的数据，约有600条用户、9000条用户信息

第6章 总结与展望

6.1论文工作总结

当今飞速发展的互联网行业，带来的是大量数据的泛滥，在这个信息爆炸的时代，用户很难从海量的信息中找到自己真正感兴趣的东西，内容的提供商也难以将自己的生成信息推荐到真正感兴趣的用户面前，即我找不到我想要的东西，我想要的东西找不到我，这是一个互相选择的过程，唯一的区别在于，对于生产者，被选择意味着有经济效益，可以将经济效益用于再生产达到企业的发展过程。作为协同消费者（用户）与生产者（内容提供商）之间互相选择的中间服务，推荐系统也由此发展，通过不断学习用户的行为数据，挖掘用户的喜好；同时分析物品的相似性，归类物品分类，为物品找到合适的买家，推荐系统往往是由内容提供商来开发生成，为了提高用户对内容的喜好，利用本平台的数据来进行推荐，增强用户对平台的依赖性，提高平台的市场占有率。其中在各种推荐系统的算法中，协同过滤推荐算法已经被成功应用在亚马逊、YouTube、Hulu等知名网站中。

6.2 论文展望

参考论文

1. 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.
2. 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.

致 谢

（小二号黑体，上下各空一行）

主要内容是表达作者对毕业设计（论文）工作的心得体会，以及对给予各类指导和协助、资助完成毕业设计（论文）工作和提供各种条件的单位及个人表示感谢。300字左右。

……

（小四号宋体，1.35倍行距）

5

附录示例

附 录

（小二号黑体，上下各空一行）

对需要收录于论文中且又不适合书写于正文中的附加数据、资料、详细公式推导、计算机程序等有特色的内容，可做为附录排写。

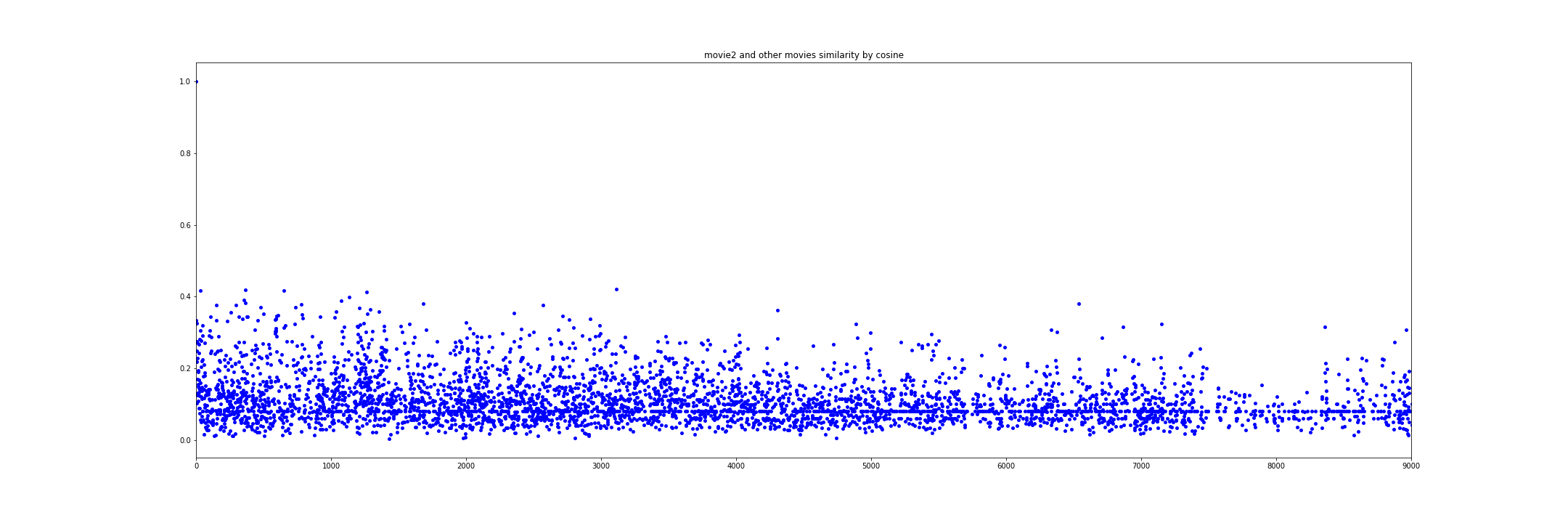
……

（小四号宋体，1.35倍行距）

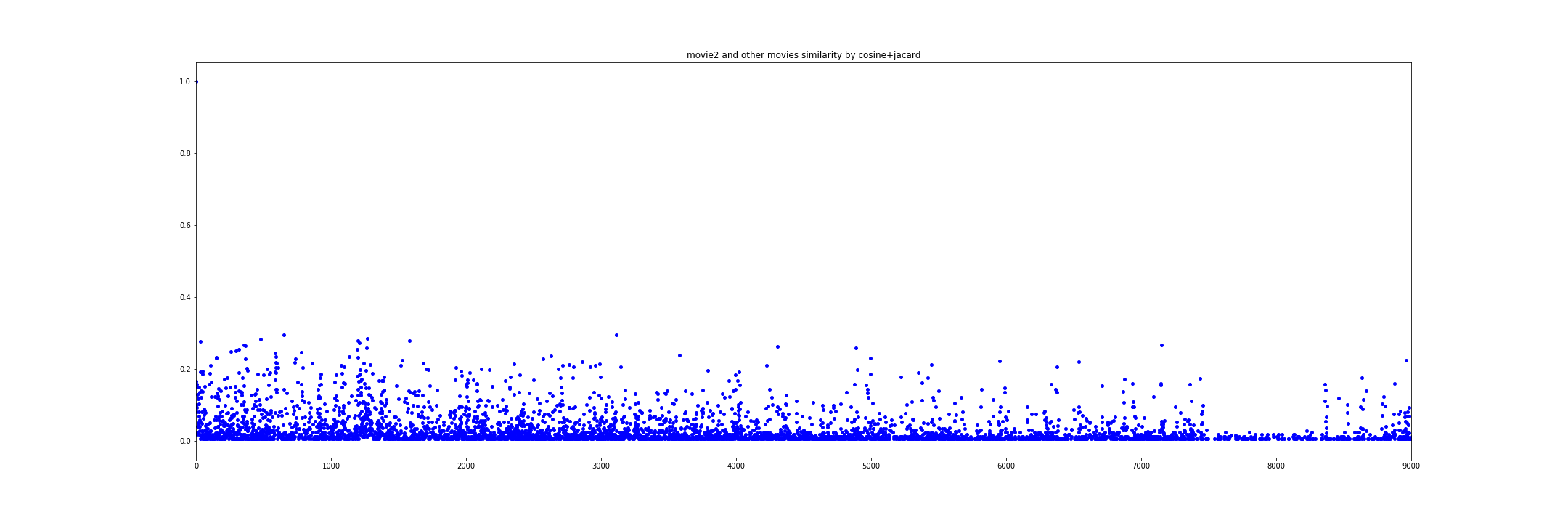
6

# itemCf-jacard_1

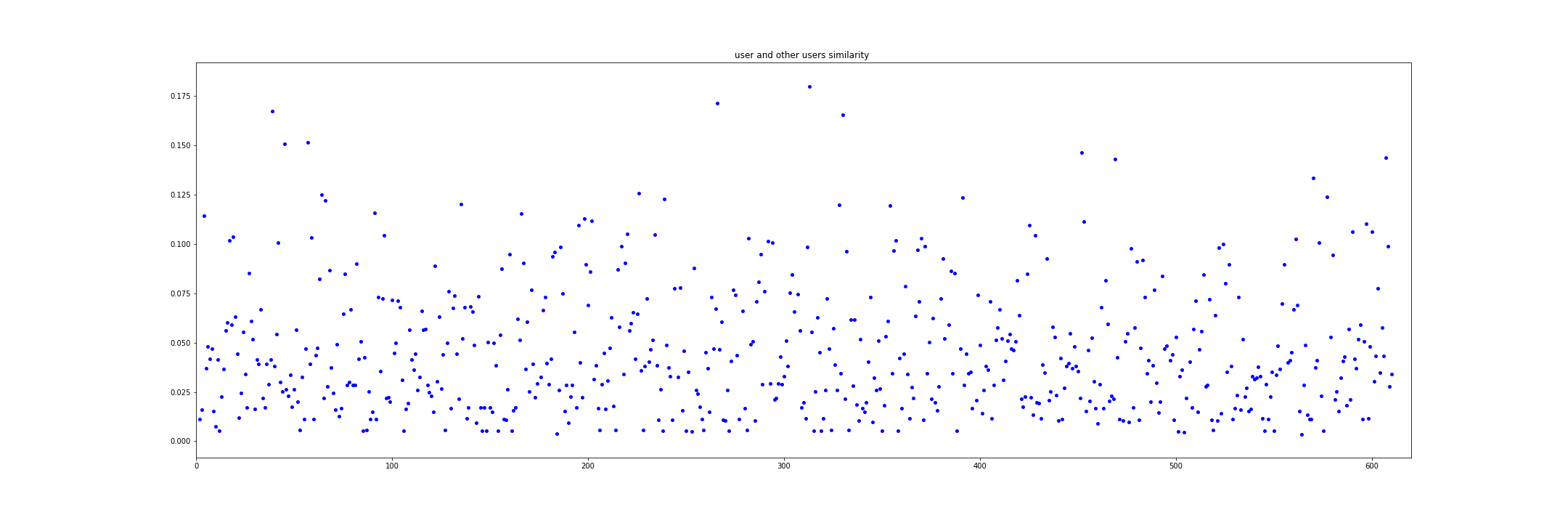
电影1与其他用户相似度（杰卡德相似度）



电影1与其他用户相似度（余弦相似度）



电影1与其他用户相似度（交集的余弦相似度\*杰卡德相似度）



用户1与其他用户的相似度