

**《软件项目管理》课程项目报告**



题 目  **图书推荐系统**

学 院  **软件学院**

专 业  **软件工程**

学生姓名  **梁爽**

学 号  **2018141463071** 年级 **大三**

二Ο二一 年 六 月

1 绪论

**1.1 项目背景**

随着计算机信息技术和互联网技术的发展，从之前的信息短缺时代，跨越到了信息过剩时代。在这种背景下，人们越来越难以从许多信息中找到感兴趣的信息。相对来说，对于信息来说，想要找到对自身感兴趣的用户也越来越难。而本文研究的推荐系统任务，就是将信息与用户连接。

想象一下，用户想要购买一本书，用户只需走进书店并按照书名直接购买即可。也可以通过淘宝、京东、当当直接搜索，进行购买。不过，这种方式的前提是用户需要明确自己的需求，确切地知道自己想买的哪本书。

但是，如果用户没有明确的目标，比如寻找自己喜欢的音乐，用户可以使用预先定义的类型或标签搜索有趣的音乐，但面对大量的音乐，实际上用户可以找到自己感兴趣的音乐。在这个时候，需要分析用户已收听音乐自动工具，用户有兴趣向用户推荐音乐。这是个性化推荐系统的工作。

信息过滤系统具有以下两个特点:

1)主动性。从用户的角度来看，门户站点和搜索引擎是解决信息过载的有效手段，但它们需要提供明确需求的用户。如果用户不能正确地说明自己的需要，则这两种方法不能为用户提供正确的服务。用户不需要提供特定的需求，但信息可以由用户推荐。

2)个性化。推荐系统的核心内容是找到长尾信息[2]。销路好的商品一般表示大多数用户的兴趣，而冷门商品一般表示少数用户的个性需求。在电子商务平台的时代，冷门商品的交付甚至超过了爆品。长尾信息的发现是推荐系统的重要研究方向。

现在，推荐系统的思想和算法已经趋于成熟，在很多领域被广泛应用，最普遍的是电子商务。同时，随着机器学习和深入学习的发展，工业界和学术界热衷于研究这一挑战性的学科体系。

**1.2 国内外研究现状**

推荐系统是一个先进的思想，所以具有其独特性，因为它是一种仅属于网络时代的个性化信息检索工具。随着互联网和大数据时代的到来，人们逐渐意识到，所有信息的不定向推广是耗时费力且收效甚微的，这就体现出来推荐系统的价值。经过20多年的积累和沉淀，它逐渐成为一个独立的问题。

1994年，明尼苏达集团透镜研究小组推出了第一个自动推荐系统， GroupLens。提出协同过滤是推荐系统中的一项重要技术。

推荐系统(recommendersystem，RS)于1997年提出。由于推荐系统一词的广泛应用，推荐系统成为一个重要的研究领域。

1998年，Amazon.com推出了一种基于项目的协作过滤算法。

2003年，Amazon Linden等人本文提出了一种基于物品的协同过滤算法。据统计，推荐系统的贡献率在20%到30%之间。

2005年，Admavicius等人论文分为三大类:基于内容的推荐、基于协同过滤的推荐和混合推荐，并提出了今后的研究方向。

2006年10月，北美在线视频服务提供商Netflix，举办了一个比赛，在学术界和工业界引起了相当大的关注。奖项丰厚，与会者提出了几种推荐算法，以提高推荐的准确性，极大地促进了推荐系统的发展[3]。

2007年在美国举行的第一次ACM推荐系统会议是2017年第11次。这是推荐系统领域的顶级会议。它提供了一个重要的国际论坛，展示不同领域推荐系统的最新研究成果、方法和方法。

在2016年，YouTube的宣布使用推荐系统深层神经网络来获得大规模的建议最有可能的建议。

近年来，推荐系统已广泛应用于电子商务推荐、广告定向投放、时事新闻推荐、抖音、知乎等平台。

经过20多年的贮存和沉淀，推荐系统在许多领域的应用取得了成功。最常见的应用场景是电商、广告、视频、社交和音乐。这些应用及需要通过推荐系统进行进一步的发展，所以这也是推荐系统研究和应用的重要实验场景。

随着推荐系统的发展，用户逐渐接受了这种模式，经过调研，用户目前不仅对模型用户历史行为的分析感到满意，而且认可了混合推荐模型。各行业的应用都在致力于通过不同的推荐方法解决冷启动和非常稀疏的数据问题。当前，中国著名新闻客户的头条新闻使用内容分析、用户标签、评级分析等方法创造了数百万美元。用户推荐引擎发展迅猛。

移动互联网的普及为移动电子商务数据、移动社会数据和地理数据等推荐系统提供了更多的数据。它成为社会推荐的新的尝试。

通过对推荐系统的应用，推荐系统的有效性评估，稳健性和安全性的算法进行了研究。在2015年，艾伦说和其他人在雷克斯会议上发言。同年，Frank Hopfgartner等人讨论了基于流数据和比较实验的离线评估方法，并进行宣布。

近年来，机器学习和深度学习的发展为推荐系统提供了方法论指导。2016年以后，RECSYS会议召开了关于推荐体系的深入学习研讨会，推动了研究，鼓励在深入学习的基础上应用推荐体系。

2017年，Alexandros Karatzoglou等人他的论文介绍了推荐系统的深度学习应用。

**1.3 项目的主要工作**

该图书推荐系统致力于为收集图书信息来为用户推荐最相关的书籍。对于用户，通过获取数据库用户信息、借阅记录、图书信息等原始数据，快速分析出用户兴趣图书的集合，完成个性化推荐功能。对于管理者，该系统能完成读者查询个人信息、图书评分检索等功能。

**1.4 项目组成员及分工**

|  |  |
| --- | --- |
| 姓名 | 分工 |
| 梁爽 | 项目团队管理，后端开发 |
| 黄英伦 | 分析新功能、系统测试 |
| 杜家琛 | 需求分析、数据库设计 |
| 杨祥睿 | 前端设计编写、模块规划 |

2 相关技术介绍

**2.1开发平台**

**2.1.1系统开发环境介绍**

Python是一种目前广泛使用的语言，非常受使用者们的欢迎，因其自身的优越性很快就得到了迅速的发展。这对C++，java等今年来流行的语言造成了很强的影响。拥有良好的通用性，作业迅速，良好的跨平台和稳性是Python技术的优点，目前，像个人电脑、数据操作、电脑和手机游戏后台、手机移动端和计算机应用的很多方面都在使用Python作为开发语言，技术稳定成熟,缩短开发时间，重复性好，在线扩展方便。Python因风格简洁、可读性较好深受编程人员的喜爱，并得到了广泛运用。对于推荐系统来说，开发过程中会遇到各种各样的问题，所以在编写程序和运行代码的过程中，关于设计的逻辑和设计过程，都是一个不断发现问题、解决问题的过程，不断完善以达到预期的功能才是我们希望看到的。Python语言具有其他语言所没有的特性，可以使用它进行开发本图书推荐系统平台。

Python的多样性,意味着可以横跨多个领域,绝不仅限于Web开发、桌面程序、移动应用,甚至包含硬件开发等。所以并没有被束缚在单一的平台之上,Python具有良好的可移植性，在图书推荐系统开发上使用Python可以大大方便项目的开发和维护。

由上面论述可知，通过Pygame工具的应用，可以简化项目的开发，Pygame使开发者不必过多的在意一些琐碎的问题，因为这些问题Pygame工具会帮我们解决，这让开发者拥有更多的时间和精力放在系统开发的关键部分，给开发者带来了相当大的便利。最关键的是，Pygame拥有跨平台的特性，这使得基于它开发的项目可以自由的在各个操作系统上运行，这省去了很多繁琐的修改。随着Pygame开发工具的流行，该工具以其独特的优越性，越来越多的被应用于各种系统平台的开发。

当前程序是以python为编程语言，主要功能实现依赖于pygame模块，主要用到surface对象之间的位置变化，再利用事件监听让程序运行起来。运行中Surface对象的位置发生变化后，界面刷新，用户对鼠标与键盘进行操作时，监听操作完成相应事件。

软件开发方式：①系统总体设计②系统详细设计③编码④测试。

系统运行环境：Windows7及更高版本。

**2.1.2 数据库系统介绍**

MySQL是一个开源的关系数据库管理系统（RDBMS），它使用广泛使用的结构化语言（SQL）进行数据库管理。

MySQL是一个开源的，因此任何人都可以在通用公共许可证下下载并更改设置以满足个人需求。

MySQL速度快、可靠性高、适应性强，一直受到人们的广泛关注。大多数人认为MySQL是在不进行事务处理的情况下管理内容的最佳选择。

因此，对于简单的数据库使用需求，我选择MySQL作为数据库管理工具。

**2.1.3 开发工具介绍**

PyCharm是一种Python 集成开发环境，它有一套工具可以帮助用户提高开发python语言的效率，例如调试、语法突出显示、智能提示、自动作业、单元测试、project管理、代码跳跃、版本控制。

编码支持：其支持智能化的 、可配置的编辑器提供代码完成、代码片段、代码折叠和窗口拆分支持，使用户快速便捷的完成任务，节省了用户的时间。

项目代码导航器：这个IDE快速的帮助引领用户在文件之间切换，从一种方式浏览语句、用法和类的层次结构。如果用户记得使用系统默认提供的快捷键或者是他们自己设置的快捷键，使用效率会更高。

代码分析：用户可以使用编码规则、错误突出显示、智能检测和一键代码快速完成建议来优化编码。

Python重构:在程序编写过程中，导入域/变量/常量，重命名，提取方法/超类，移动和前推/后退重构这些操作可以使用这个功能来实现，极大的减少了用户的任务量。

集成版本控制：如果用户想使用这些功能，如登录、输入、视图拆分和合并， -用户可以在VCS用户界面中找到，这是其通用的功能。

具有自带的调试器，调试器的功能多样化，可以提供多种功能，用户通过对基于python和 Django的项目进行调试，同样，系统的单元测试，也可以通过它来解决，该调试器包括blake点、分步、多屏幕视图、窗口和计算表达式等。

集成单元测试：用户可以运行测试文件，单个测试类。一个方法或者所有测试项目。

另一方面，Pycharco还为Django的开发提供了一些很好的功能，以及对Google应用引擎的支持，以及对Pycharm的支持。

**2.2开发技术**

python用作该软件的开发语言，其关键技术在于布局、事件监控和数据存储。布局主要是美化界面以及界面的排版。用户交互界面用布局来实现，给用户带来美观、舒适、直接的用户体验。所以，界面布局的设计会影响到使用者的客观感受，获取用户操作使用事件监视技术，而数据监视技术记录所有操作，创建用户需要提取的数据。

Python是一种优雅、简单、健壮的开源解释语言。产生于1989年，由Givavo RSM开发和设计，设计Pyhlo的最初目的是为了高效的完成某一项任务而创造的。它从一种为提高研究项目的工作效率而创建的通用编程语言开始。经过多年的发展，python已经逐渐得到改进。由于其强大的可扩展性和广泛的库支持，它已经出现在许多领域，如豆瓣等就是成功的应用python技术的例子。

Python的主要特点有：

l）低入门标准的python语法相当简短，编写的程序通常简短，非常像日常使用的自然语言，有利于开发者的使用和理解。

2）Python是一门面向对象的语言，在面向对象中，与面向过程语言的差距就是类和对象的使用，体现了python的特征，面向对象的语言特点是在该语言被创造的时候就体现的。Python之所以能成为一门被大众喜爱的编程语言在于它的精心设计的数据和内存管理。

3）对于内存的管理，python也有其特定的部分负责，对于开发者来说，程序就显得有必要了，因为开发者只有理解这些程序的前后逻辑才能更好的写出项目，使他们不像C/C++程序员那样专注于处理内存事务。Python的程序设计和编写时间更短、出错更少也是基于此特性。

4）主机语言与其通信可以方便的被嵌入，可以用C语言编写对于一些对性能特别强调的地方，这些扩展在python中被调用以实现性能改进的目的。相反，Python解释器可以嵌入到C/C++中，它取代接口可以通过动态链接库的形式进行，通过这种方式，程序开发由此变得灵活方便。

5）在python的标准库中，含有多个模块来实现具体的功能，这些几乎包含了所有与操作系统解释器的交互的功能，也就是Python使用者不用手动人工造轮子，因为这些模块可以直接用于已经完全测试过的功能开发。这些已经被充分测试的模版在实际的编程开发中得到了充分的应用。

**2.3 关键算法**

**2.3.1 常见的推荐算法**

现如今网上信息泛滥，想要在里面找一条适合自己的信息的成本真的有点高，如果可以较为完善的推荐系统出现的话，于用户而言，可以大大的节省自己的时间；从商家的角度来看，通过推荐系统可以更为精准地投放自己的商品对象，从而可以更好的卖出自己的商品。

根据使用数据源的不同可将其大致分为三类：

1）协同过滤的推荐方法

2）基于内容的推荐方法

3）基于知识的推荐方法

这三类算法通过不同程度融合，可以出现混合推荐算法。

一、协同过滤算法，其中包括基于用户的协同过滤及基于物品的协同过滤。

1）基于用户的实现原理：

1.计算用户之间的距离

2.将用户之间相近的，推荐给他们喜欢的物料

3.通过收集用户反馈数据，进一步优化用户之间的距离

2）基于物品的实现原理：

1.计算物品之间的相似度矩阵

2.收集用户评分高的物物品

3.将与用户评分高的相似度较高的物品，推荐给用户

4.通过收集用户反馈数据，进一步优化数据

二、基于内容推荐算法

实现原理:建立用户画像-行为偏好，建立物品画像，特征，通过相似度计算，然后推荐。

弊端：建立用户画像，需要基于大量用户行为数据。

三、基于知识推荐算法

实现原理：基于知识的推荐算法主要将重点放在知识源，没有冷启动的问题，因为推荐的需求都是被直接引出的。其主动的询问用户的需求，然后返回推荐结果。

弊端：“知识”的获取比较难。

**2.3.2基于物品的协同过滤算法**

协同过滤推荐算法是推荐系统中最基本的算法。它分为基于用户的协同过滤算法（usercf）和基于物品的协同过滤算法（itemcf）。

基于物品的协同过滤算法主要分为两个步骤。

1）计算物品之间的相似性。

2）根据物品相似度与用户历史行为的，向用户提供推荐列表。

第一步骤中的关键点是计算项之间的相似度。除了使用基于内容的相似性，它是计算有多少类似的物品，而是看喜欢i的用户中，有多少人喜欢j的，因此计算是基于用户。该兴趣一般都比较确定和不容易改变。当一个用户都喜欢的物品，我们通常可以认为，这两个物品可能属于同一类别。令N(i)表示购买物品i的用户数，则物品i和物品j的相似度可以用公式1来计算。

第一步的时间复杂度的改进方法：以UserCF类似，我们可以创建一个用户，项目查找表，通过计算，认为用户有beenhave这些项目之间的相似性时，它可以保证计算的相似性。这样能够保证相似度是有用的，而不用对那些零（可靠地稀疏矩阵）花费大量的计算量。

第一步相似的改进方法1：如果按上述公式计算的相似性，可以发现，这个物品我和受欢迎的物品j之间的相似性是非常高的，因为流行的读数偏高，所以基本上每个人都会买它。具有较高的知名度的商品不太区分的，所以我们需要惩罚流行物品j的权重。

第一步相似性改进方法2：需要惩罚用户的活动。如果用户不活跃，只有购买的图书数量有限，那么这些书很可能在计算项目中感兴趣的一个或两个区域的相似性是有益的，但如果一个书店卖家提供折扣，如果你买90%Amazon的书籍，然后赚取差价，那么用户的行为对计算物品的相似性不会有任何作用，因为90％的书肯定会涵盖了很多的范围，所以你应该惩罚用户的活动，可以采取第一个方法。

第一步相似性改进方法3：物品的相似性的归属。规范化不仅提高了建议的准确性，还增加了建议的覆盖范围和多样性。例如，在京东上，用户的爱好种类繁多，有相机爱好者、耳机爱好者、电脑爱好者等。很少有人说爱好集中在一个类别中。假设有两种类型的A和B.A类之间的相似性是0.5，B类之间的相似性是0.8，A和B之间的相似性是0.2。当用户购买A类的5本书和B类的5本书后，我们必须向用户提供推荐。如果我们按照前面的方法并按相似性排序，那么推荐的方法应该是B类项目。即使B类别较低，它仍然优于A类。为了高相似性，所以相似性的相似性应该基于类别，因此A的相似性为1，B的相似性也为1 ，以便排序后推荐的A和B产品具有更高的准确性，覆盖范围和多样性。

第二步则比较简单，计算物品与用户已买物品的相似度（权重和），然后根据相似度排序选出topN。

ItemCF在实际系统中运用的比较多，主要有两个优点：

1）item-item表相比如user-user表要小的多，处理起来比较容易

2）itemcf很容易为推荐提供理由，比如提前进行数据挖掘，可提高可靠性，改善用户与推荐系统的相互作用，并进一步加强定制推荐前推荐数据挖掘等。

基于物品的协同过滤算法与用户的协同过滤算法相比，基于用户的协同过滤算法有两大缺点。

1）随着网站用户数量的增加，计算用户数量的相似性就更加困难了。计算的时间复杂度和空间复杂度与用户的增长基本成平方关系。

2）基于用户的合作过滤算法很难对推荐的结果作出解释和建议。

基于物品的协作过滤算法是“目标用户”，用于查找与其喜欢的项目类似的项目。从实际情况角度出发，在对于本课题中的需求——图书推荐来说，每个用户对于个性化推荐书籍的需求都比较强烈，此时采用基于物品的协同过滤算法就可以更为充分地挖掘用户的兴趣领域，并且很容易根据用户的历史数据来对推荐结果做出解释，从而使用户更加信任系统做出的推荐结果。

3 设计工作和进展情况

**3.1.1 需求分析与建模**

需求分析是软件工程中的一个重要步骤。这个阶段的主要任务就是调查用户需求。并和开发人员进行确认，将客户的非技术性需求转化为技术上可实现的技术性需求。明确要实现哪些功能、完成哪些工作，产出规范性文档《需求规格说明书》。在通过评审后，《需求规格说明书》起到了桥梁的作用，成为用户、开发人员进行理解与交流、反映用户的问题结构以用作软件开发的工作依据、作为软件测试和验收的依据。

总体设计原则的提出是为了确保系统建设成功，并为系统的可持续发展做出规划。因此在系统设计时，我们遵守以下原则：

简单性：实现系统需求的前提下，尽可能的保证系统简单易操作。一方面，简单的操作会使增进用户体验，另一方面操作过于复杂时，更易引入问题及漏洞。

针对性：本项目是选用基于物品的协同过滤算法实现图书推荐系统，用明确项目需求为目标，具体需求具体实现，有很强的针对性。

实用性和一致性：具有较高的视觉一致性。主界面采用tkinter模块进行编写，主界面上的元素统一使用标签插入，具有较高的功能一致性。

**3.1.2 可行性分析**

可行性分析包括技术性、经济可行性、社会可行性及其它考虑。

为避免浪费投资，提高软件生产的成功率。这是解决问题的实际目的，问题可以短时间以小成本解决。

下面对基于基于物品的协同过滤算法实现图书推荐系统开发进行判断和考察，主要从技术、经济、社会几个方面来分析。

技术可行性：

主要分析技术要求的技术可能性可以完成开发任务，硬件和软件可以满足开发人员的需求。功能强大的JetBrains是该软件中使用的开发工具。

PyCharm 2018.2.3 x64，强大的扩展能力是该软件的特性，该软件对于系统编写及完善有良好的支持效果，也是众多Pythoner喜爱的编译器。随着互联网行业的迅猛发展，软件开发平台及硬件技术同时不断更新进步。大容量、可靠性的提高、低价格也使得软件开发是可行的。本系统的编译需求完全可以由Pycharm编译器承担。

综合以上情况及考虑，本系统的开发在技术上是完全可行的。

经济可行性：

由于本系统较为小型轻便，开发成本较低。此外，该软件稳定、后期维护简单、实用，一旦开发完成即可长期使用。当用户有了新的需求时，只需要根据需求，在原有代码基础上进行更改，维护成本较低。

综合以上情况及靠背，本系统在经济上是完全可接受的。

社会可行性：

法律因素：本系统是本人处于兴趣爱好，独立完成开发的。基于Python完成，同时借鉴市场上同类软件的功能，收集并归纳用户需求，制订设计思路，结合实际中存在的实体，进行创新及开发的。

用户使用可行性:

本系统对用户的要求，没有复杂繁琐的操作，简单易用。使用软件的用户， 在了解了简单的流程后就可以对后台进行管理，没有额外的学习使用环节，节约成本。

由以上分析可知，本系统在社会可行性方面是完全可行的。

**3.1.3系统组成**

分为硬件和软件部分，硬件部分需求足够高的性能以及内存和磁盘空间用作系统运行和部署，软件部分则分为数据库，后端以及前端部分，数据库用于储存各类数据，后端用于接收衔接前端和数据库，用于处理需求和数据，前端用于可视化操作和输入操作。

**3.2 项目开发计划及阶段性完成情况**

**3.2.1 工作内容**

项目目标**：**

1、建立图书分类系统

2、通过评分、简介等记录进行图书协同过滤

3、加入个性化图书推荐功能

项目范围：

前端与后端开发，数据库开发，数据库与系统的连接，调试与成品的测试使用。

需完成的软件：

完成程序名称：图书推荐系统

程序内容：源程序、数据库对象创建语句、可执行程序、支撑系统的数据库数据、配置文件、第三方模块、界面文件、界面原稿文件、声音文件、安装软件、安装软件源程序文件等等。

所用编程语言：Python

软件开发方式：

1）系统总体设计

2）系统详细设计

3）编码

4）测试

需提交用户的文档：

需求规格说明书，概要设计说明书，详细设计说明书等。

需提交内部的文档：

软件项目开发计划书，设计文档，项目报告等。

### 项目开发环境：

操作系统：windows10

开发工具：PyCharm

数据库系统：mySQL

项目验收：

项目完成后首先由开发小组内部进行测试和验收，确定无误后进行演示，随后结束项目开发验收的整个流程，项目验收依据为项目文档，如项目开发计划书、需求文档、设计文档和详细设计说明书等。

**3.2.2 工作进度安排**

本次项目的截止时间约为五月中旬，整体开发时间约为八个周，项目小组成员为四人，使用设备为笔记本电脑。

组长：负责项目开发进度的跟踪，开发任务的分配，进行项目总结，解决人员问题

前端开发：负责项目的前端开发

后端开发：负责项目的后端开发

数据库人员：负责数据库部分的具体实现

测试人员：负责对已经完成的代码部分进行测试反馈

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 起止时间点 | 所需资源 | 完成工作 | 应提交成果 | 检查点 |
| 2-3周 | 相关的需求和开发数据 | 项目计划和需求分析 | 项目需求报告和项目开发计划书 | 完成报告和计划书编写 |
| 3-4周 | 相关开发经验和工具 | 项目开发前期准备 | 开发环境配置和相关工具和数据的准备 | 完成前期准备 |
| 4-6周 | 开发人员和相关技术支持 | 大致完成项目整体的编写 | 已经完成的项目初期版本 | 项目大体编写完成 |
| 7周 | 测试人员和相关测试经验支持 | 完成后期收尾和测试等工作 | 项目的最终成果 | 开发结束，完成项目测试 |
| 8周 | 开发流程数据汇总 | 项目汇报 | 项目成品 | 完成项目汇报 |

**3.3 项目的开发内容和结果**

**3.3.1 系统需求分析**

**3.3.1.1 功能需求**

图书推荐系统分为两个子系统，后台管理系统和客户端系统。后台管理系统由管理员操作执行，客户端由读者操作执行。

一、**后台管理系统：**

1.用户管理

2.信息导入

3.热门图书管理

4.分类热门图书管理

**二、客户端：**

1.读者账户管理

2.读者信息管理

3.个性化推荐

4.热门排行

**3.3.1.2性能需求**

1.普通情况下，用户从操作到得到系统反馈的时间最大不超过5秒，平均时间在1～3秒以内。

2.数据库至少能储存10000条数据记录，以确保能够录入足够大小的用户的阅读信息与图书量，保证程序正常运行。

3.数据库服务器能够承载至少1000人同时进行操作。

**3.3.1.3 接口需求：**

用户界面需求：

程序将采用1920×1080分辨率的窗口模式运行。

**3.3.1.4 其他需求：**

1.安全性

2.高可用性

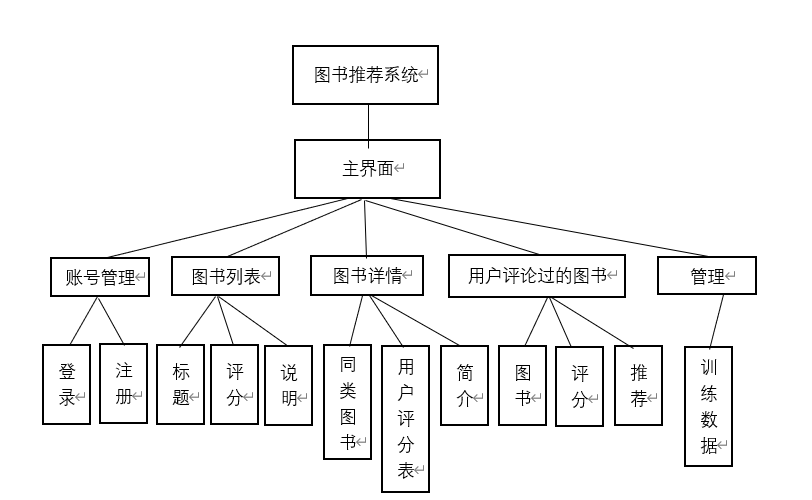
3.易用性

4.高性能

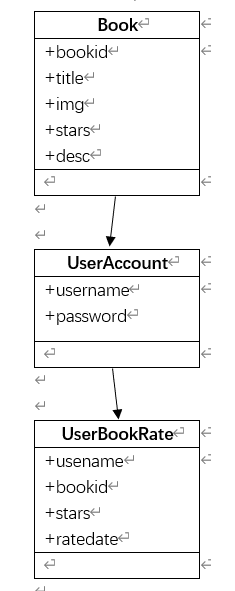
**3.3.2 系统设计**

**3.3.2.1体系架构设计**

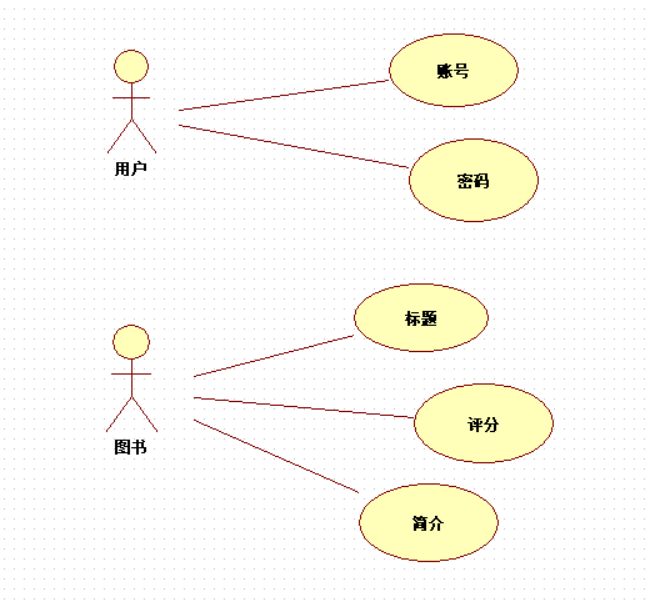
**功能模块图：**

****

**类图：**

****

**用例图：**

****

**硬件平台**

办公用笔记本电脑。CPU：i3-530以上，内存2G以上，硬盘容量足够大。

**软件平台**

windows 7及以上操作系统，IE6.0版本及以上浏览器。

开发环境：Python

数据库系统：MySQL

开发工具：PyCharm

**系统可靠性设计**

普通情况下，用户从操作到得到系统反馈的时间最大不超过5秒，平均时间在1～3秒以内。

系统能够保护用户信息，包括用户账号密码等。且能阻止外部侵入，防止数据库信息泄露或被窃取。

系统能够确保在一年内最多因故障停机5次，且每次停机时间不超过2小时。

系统能够在多个并发读写中保持稳定，保证数据库服务器不会崩溃。

**3.3.2.2人机界面设计**

**1、登录与注册**

登录与注册设计采用BootStrap的Navbar导航条，插入Button和Entry对象来创建按钮及输入框，并设置了绑定变量来获取输入框输入。

具体实现步骤为:

a.设置导航条，其中包括图书推荐系统、图书列表、用户评价后的图书、管理。

b.登录部分设置两个标签，分别命名为“用户名”、“密码”，放置两个输入框，两个按钮“登录”和“注册”。注册部分设置三个标签分别为“用户名”、“密码”，“确认密码”放置三个输入框，一个“注册按钮”。

c.密码的文本设置输入密码后显示\*号。

**2、图书列表部分**

在该部分，选择Web API类型的MVT，使用Django是一款python的web开发框架：与MVC有所不同，属于MVT框架。

m表示model，负责与数据库交互。

v表示view，是核心，负责接收请求、获取数据、返回结果。

t表示template，作用就是将内容反馈并呈现在浏览器上。

通过与数据库的交互，获取数据并返回结果到图书列表的界面上。

首先获取到图书的图片，通过book.title获取标题并将其显示在界面上，通过book.stars获取评分情况将其显示。通过book.desc得到图书简介。

**3、图书详情部分：**

这部分最关键的部分是显示与此书相似的图书，对训练的结果中的相似图书按照一定的格式显示在页面上。判断是否曾经对其进行评分，如果评分过，显示之前的评分。反之，提示用户“您还没有评分过，给个评分吧”

**3.3.2.3系统数据库设计**

根据系统业务和DBMS的需求，建立了最佳的数据存储模型。另外，通过建立数据库内的表结构与表与表之间的关系的处理，能够有效地将数据存储到应用系统中，高效访问存储的数据[15]。好的数据库设计需要以下部分。

减少数据冗余；避免数据维护异常；节约存储空间；高效的访问；需求分析。

同时需要分析数据和属性各自的特点，以便了解系统中所要存储的数据、了解数据的存储特点、了解数据的存储周期。需求分析中需要了解的问题是实体之间的关系、包含的属性。

该系统采用MySQL数据库，保存用户的用户名、密码等数据。

数据库的主要表如下：

用户（账号，密码）如表1所示。

图书（标题、评分、简介）如表2所示。



**3.3.3 系统具体实现**

**Python实现核心功能**

**3.3.3.1 跳转方式**

当用户在aw和bw未登录时，在SSO上设置登录态，那么在aw和bw上应该设置登录态。如上所述，还是应该在aw和bw上设置各自的登录态，这样在访问aw时首先会在aw域上检测授权，如果没有授权，则跳转到SSO进行登录授权。但是aw和bw应该为登录态一般设为浏览器进程存活期，即aw和bw的登录态的存活期直到浏览器关闭。SSO域上登录态的存活期取决于具体的业务，本系统中设为7天。代码如下：

app = Flask(name)

app.config[‘SECRET\_KEY’] = os.urandom(24)

# 服务器启动一次上次的session就清除，因为设置为随机产生的24位的字符，也就是说每次运行服务器都是不同的。

app.config[‘PERMANENT\_SESSION\_LIFETIME’] = timedelta(days=7)

# 设置session的保存时间。

**3.3.3.2 数据库连接**

首先创建数据库连接，打开数据库连接。代码如下：

import pymysql

def create\_connection():

db = pymysql.connect(“localhost”， “root”， “12345678”， “bookrecommend”)

return db

**3.3.3.3 ItemCF算法的实现**

1.ItemCF算法的实现是本图书推荐系统的核心内容，首先要计算出物品之间的相似度，相似度是推荐的关键评判指标，然后根据物品的相似度，并结合用户的历史行为，即那里评分矩阵，从而给用户生成可靠的推荐列表。

具体步骤：

首先建立物品的同现矩阵。

其次建立用户对物品的评分矩阵。

最后矩阵计算推荐结果。

def ItemSimilarity(train):

# 物品-物品的共同矩阵

C = dict()

# 物品被多少个不同用户购买

N = dict()

for u， items in train.items():

for i in items.keys():

N.setdefault(i， 0)

N[i] += 1

C.setdefault(i， {})

for j in items.keys():

if i == j:

continue

C[i].setdefault(j， 0)

C[i][j] += 1

2.计算问题之间的相似度

W = dict()

for i， related\_items in C.items():

W.setdefault(i， {})

for j， cij in related\_items.items():

W[i][j] = cij / math.sqrt(N[i] \* N[j])

return W

问题i与问题j之间的相似度这么定义：同时关注问题i与问题j的人数/关注问题i人数关注问题j的人数的平方根。

由此我们就需要计算n个问题之间两两相似度，就是一个对角线为1的对称邻接矩阵，也就是说，想要成功求出n个问题之间的相似度，至少需要计算n(n-1)/2次，这样才可以计算n个问题之间的两两相似度。

推荐前K个用户：

def Recommend(train， user\_id， W， K):

rank = dict()

if user\_id not in train: return []

action\_item = train[user\_id]

for item， score in action\_item.items():

for j， wj in sorted(W[item].items()， key=lambda x:x[1]， reverse=True)[0:K]:

if j in action\_item.keys():

continue

rank.setdefault(j， 0)

rank[j] += score \* wj

return sorted(rank.items()， key=lambda x:x[1]， reverse=True)

**3.3.4 系统测试**

测试就是为了寻找程序中的错误，一个成功的测试就是发现至今尚未发现的错误的测试。在这样一个找错误的过程中，每发现一个错误都是值得高兴的，因为这可以又为以后排除了一个潜在的隐患，同时系统如果要是经过测试投入使用后再发现系统的错误，那么将会花费更多的物力和财力才能解决问题，这样无疑会造成投入成本增加，更严重的甚至导致该系统不能使用白白浪费了之前的投入。因此在系统正式投入使用之前，尽可能全面的去测试系统无疑是必不可少的，也是非常重要的。

在实际的生产生活中在真实的系统工作环境下通过与之前的需求分析进行作比较，检查实现的功能是否符合需求分析的要求，是否满足客户要求。系统测试主要有两种常用的方法，黑盒测试和白盒测试，它主要是根据是否关心程序的内部结构来划分的。所谓的黑盒测试是指不关心程序如何编写的，内部结构如何，只关心输入和输出的结果是否的正确。白盒测试是将程序看成一个透明的盒子，进而分析程序内部的运行情况是否正确。

测试就是通过静态审查或者运行程序从而找出软件实现过程中是否与需求存在偏差。除此之外，测试还需要软件质量进行度量，这样可以全面掌控软件情况，从而及时采取措施推进问题修复。测试的原则之一是测试应当尽早介入，这样的话，在研发阶段，越早发现问题，造成的损失越小，所以该问题的修复成本就越低。在寻找错误的过程中，不断完善系统，尽量的避免问题出现与系统投入使用过后，修复成本大幅度增加的情况。因此，进行全面的系统测试是不可或缺的。

在实际的测试流程中，将在设计阶段根据需求撰写测试用例，并根据实现情况对用例进行补充及修改，验证需求实现情况，提出问题并推进修复，最后给出软件质量度量。

执行程序给定输入，并校验程序输出是否符合预期结果是黑盒测试的标准流程。它不必关心程序的内部实现逻辑，不需要涉及代码层面[2]。与之对应的是白盒测试，大多是通过静态代码审查的方式，不运行程序，凭借程序实现的逻辑结构，来检查程序是否满足需求，符合预期。

**3.3.4.1测试的定义**

软件测试是直接影响软件质量评价的重要部分，这是用来衡量实际开发的系统与预期结果之间的审核比对过程，查缺补漏。软件测试需要采用有效的方法，及时发现问题，防止后续出现难以掌控的状态。测试的作用就是在衡量成本的条件下，尽可能的找出问题并推动问题修复，为软件质量作出度量，避免问题出现在系统投入使用后，减少修复成本。

**3.3.4.2测试的重要性**

随着软件行业的蓬勃发展，人们更加关注的是软件质量的实现，因此软件测试就愈发重要。在软件工程中，这是必不可缺的一环。在操作和维护阶段之前，应确保软件的质量，然后在操作和维护阶段向用户提供软件产品。随着系统测试越来越重要，测试理论、测试方法越来越科学，普遍认为：软件测试应该在项目之初就进行介入。虽然软件测试的原则之一就是测试不可能发现所有BUG，软件必定是存在BUG的。但是为了减少错误的引入，我们可以使用完备的开发过程、有效的开发方法和新的语言，但这些方法只是最大程度的降低了错误。通过测试来检测这些误差是很有必要的。BUG具有集群性，符合“二八原则”。测试过程中，通过统计问题分布，可以推算哪些模块需要继续进行测试。

**3.3.4.3 测试方法及过程**

白盒测试是基于代码的一种测试设计方法。测试人员通过审查代码、明晰代码实现逻辑后，以验证程序逻辑为目标，构造测试用例及数据。与之对应的黑盒测试，不关心程序内部的实现逻辑，只着眼于外部结构，将程序视作一个“黑盒子”，输入测试数据，查看输出结果与预期结果是否一致。运行程序、进行输入，然后得到实际结果从而验证软件是否满足需求。

探索性测试是一种测试思维与理念，在对系统有一定了解的情况下，不严格拘泥于先设计后执行的测试思路。探索性测试是设计与执行同时进行的，实践表明，探索性测试往往能暴露出更多的问题。

在本系统中，综合采用了两种方法，还结合了探索性测试的思路，使测试过程可以暴露更多问题。

**本次测试主要达到以下测试目的：**

（1）功能检查：根据需求与实现产品，验证功能需求实现情况，检查功能实现是否有遗漏偏差。

（2）数据检查：主要对用户、登录注册信息进行检查，检查数据库与程序之间的交互部分。

（3）性能检查：针对于本次的系统，该项主要是检查推荐的图书，是否存在错误现象

（4）稳定性检查：由于事件读取是从鼠标键盘上获取的，而导致后台响应频率较高，所以要检查程序是否出现运行终止、重启等现象。

**3.3.4.4各模块具体测试**

**1、登录测试**

对用户输入的用户名，密码进行验证匹配。与pymysql结合简化了SQL查询，并与pymysql.connect协同工作将使编码量大为减少。

测试编号：1

描述：本测试用例用于验证系统用户登录

前提：用户已进入系统且进入用户登录页面

备注：使用正确的用户名和密码登录系统，与数据库信息进行验证，验证成功后，必须进行登录操作的二次检查操作，即使用错误的密码登录，也需要验证系统登录验证是否准确，以及相关的错误反馈信息是否正确。

具体步骤：

1.进入图书推荐系统

2.进入主页面之后，在“用户登录”部分、在对应的文本框中分别输入用户名和密码信息。

3.单击“登录”按钮，启动系统登录确认

4.在页面“用户登录”区域，用户名文本框和密码文本框中分别输入错误或者不存在的用户名和密码。

输入值：用户名，密码

期望结果：步骤2验证通过，登录成功。步骤4准确反馈错误信息（密码错误、账号不存在）。

实际结果：步骤2验证通过，登录成功。步骤4准确反馈错误信息（密码错误、账号不存在）。

是否通过：通过

**2、注册测试**

在注册的过程中，加入校验。经测试无误。

function onregistclick() {

var username = $("#name").val();

var password = $("#password").val();

var repassword = $("#repassword").val();

if(!username) {

alert(‘用户名不能为空’);

return;

}

if(!password) {

alert(‘密码不能为空’);

return;

}

if(password!==repassword) {

alert(‘两次密码不一致’);

return;

}

$.post("/regist"， {“username”:username， “password”:password}， function (resp) {

if(resp&&resp.result){

window.location = ‘/login’

}else{

alert(‘注册失败，请更换个用户名！’)

}

})

}

测试编号：2

描述：本测试用例用于测试用户“注册账号”功能是否正确

前提：用户进入系统，在“注册账号”功能页。在“注册账号”页面编辑后提交。

备注：需要再次进入页面以确保数据已完全更新。

步骤：

1.进入图示推荐系统登陆页面，点击注册账号，进入注册账号页面。

2.在页面输入账号信息（用户名、密码、确认密码）

3.点击注册按钮，将新用户的账号数据添加到数据库中。

输入值：用户名，密码，确认密码

期望结果：用户添加账号成功。

实际结果：用户添加账号成功。

**3、数据库连接测试**

在这里，数据库管理工具用的是Navicat，选择它的原因是，全面的图形化方式进行数据库的管理，所以非常易用而且可靠，Navicat与数据库进行连接的原理是通过SSH通道和HTTP通道，既可以最大程度上避免漏洞保护数据信息，又可以在使用时，对远端服务器的访问不受安全性影响。可利用图形化界面，直接对数据库对象进行创建、编辑和删除等。

测试编号：3

描述：本测试用例用于测试数据库连接是否成功

前提：注册后的用户进入系统，通过登录确认进入推荐系统页面。在后台检查用户信息进行校验。

备注：需要再次进入页面以确保数据已完全更新。

步骤：

1.注册

2、进入图书推荐系统后，通过验证进入普通用户权限界面，首页即是图书推荐系统首页。

3.查询数据库文件看是否有新用户加入、或者是重新登录，查看是否可以成功登录。

4.输入用户名和密码，点击“登录”按钮，更新数据库中的信息

期望结果：用户成功登录。

实际结果：用户成功登录。

是否通过：通过

**4、图书详情显示测试**

测试编号：4

描述：本测试用例测试图书列表查看是否实现。

前提：用户已经进入系统，通过登录验证进入推荐系统功能页面。

步骤：

1.进入图书推荐系统的功能页面。

2.点击“图书列表”。

期望结果：用户可以查看图书列表。

实际结果：用户可以查看图书列表。

是否通过：通过

测试编号：5

描述：本测试用例用于测试图书图片显示是否成功是否正确

前提：用户已经进入系统，通过登录验证进入推荐系统功能页面。

步骤：

1.进入图书推荐系统的功能页面。

2.点击“图书列表”

期望结果：用户能够查看图书具体信息（图片）。

实际结果：用户能够查看图书具体信息（图片）。

是否通过：通过

测试编号：6

描述：本测试用例用于测试查看图书详情是否能成功显示。

前提：用户已经进入系统，通过登录验证进入推荐系统功能页面。

备注：需要再次进入页面以确保数据已完全更新。

步骤：

1.进入图书推荐系统主页面

2.点击“图书列表”

3.进入图书详情界面。

期望结果：图书详情显示无误。

实际结果：图书详情显示无误。

是否通过：通过

测试编号：7

描述：本测试用例用于测试“评分”功能是否正确，并针对不同的评论状态，查看之前的评论或者提交评论。

前提：用户已经进入系统，通过登录验证进入推荐系统功能页面。

步骤：

1.进入图书推荐系统主页面

2.点击“图书列表”

3.进入图书详情界面。

4.查看评分

期望结果：用户如果评过，提示已经评分，若未评分，则可以进行评分。

实际结果：用户如果评过，提示已经评分，若未评分，则可以进行评分。

是否通过：通过

测试编号：8

描述：本测试用例用于测试用于测试 “显示标题、简介”等功能是否正确

前提：用户已经进入系统，通过登录验证进入推荐系统功能页面。

步骤：

1.进入图书推荐系统主页面

2.点击“图书列表”

3.进入图书详情界面。

期望结果：可以正确显示标题、简介等信息。

实际结果：可以正确显示标题、简介等信息。

是否通过：通过

**5、图书推荐测试**

循环输出得到的相似图书，经测试，输出无误。

{% for book in simbooks %}

{{ book.title }}

{{ book.desc }}

{% endfor %}

测试编号：9

描述：本测试用例用于测试“推荐图书”功能是否能实现。

前提：用户已经进入系统，通过登录验证进入推荐系统功能页面。

步骤：

1.进入图书推荐系统主页面

2.点击“图书列表”

3.进入图书详情界面。

期望结果：用户可以查看“推荐图书”列表及详情信息。

实际结果：用户可以查看“推荐图书”列表及详情信息。

是否通过：通过

4 讨论与体会

**4.1 对项目过程的体会**

我作为组长主要负责协调管理和后端开发。

在开发过程中进行了选题以及开发计划的制定，在与组员的交流讨论中发现：全员共同参与，有对项目一起开发的积极性很重要，组长应该能调动组员，积极主动组织开发活动。分工明确，有计划和组织是组长应该做到的很重要的任务。

我主要负责协调过滤的后台实现，发现这个基于机器学习的算法其实有很大的优化改良空间，实现当前的系统只是完整了初步的图书推荐，在此基础上可以进行更多的算法改良和训练，算法实现是图书系统的核心，在开发过程中应该更注意代码质量和可读性，这样才能利于前后端接口的编写和测试人员的工作完成及代码的重构等。

**4.2 对项目的评价**

项目流程长，开发时间较短，在较短时间内实现图示推荐系统较困难，仅实现基础功能，未有过多的改良和优化，系统还存在许多需要改进的地方，同学的分工配合需要加强和磨合组长的管理计划分配能力需要加强和进步。

5 小结

报告首先阐述了推荐系统开发的经过，分析了推荐系统目前的研究状况。其次，阐述了推荐结果的主要推荐方法和评估指标。最后分析了主流的推荐算法以及它们各自的优缺点，尤其是基于物品的协同过滤算法。

总结项目开发流程，各阶段目标基本明确，开发人员进度较慢，总时间较短，前端开发较快，后端算法实现及训练耗费时长，测试顺利，数据库建立快速流畅。开发人员开发能力有所增强，组长管理计划能力需加强，完成文档及答辩阶段应锻炼演讲能力及临场发挥能力，提高个人编码能力。

协同过滤算法应用广泛，应加强算法的掌握理解，对算法的深度认识既有利于开发、工作也有利于研究、深造，是代码的核心部分，应加强理解与重视。

参考资料

CSDN：协同过滤算法（推荐算法）、归一化处理

道客巴巴：图书推荐系统的设计与实现

《C++ primer》[美]Stanley B.Lippman 著，王刚 杨巨峰 李忠伟改编

《计算机算法设计与分析》（第五版） 王晓东著

《数据库系统概念》(美国)希尔伯沙茨著

《Android应用开发揭秘》 杨丰盛著

《Android studio应用开发实战详解》 王翠萍著

《MySQL技术内幕：SQL编程》 姜承尧著

附录1项目开发计划

附录2需求规格说明书

附录3设计文档