一、课程设计目的

1.1 绪论

随着互联网与移动互联网迅速普及，网络上的电影娱乐信息相当庞大，人们对感兴趣的新闻信息的需求越来越大，个性化的新闻推荐成为一个热门。基于个性化推荐算法形成的产品成为人们获取信息的主要途径。网络新闻推荐系统旨在解决新闻信息爆炸的问题，为用户提供个性化的推荐服务。一般来说，新闻语言是高度浓缩的，包含了很多知识实体和常识。

个性化推荐一词最早出现在电商行业，是根据用户的兴趣特点和购买行为，向用户推荐用户感兴趣的信息和商品。随着电子商务规模的不断扩大，商品个数和种类快速增长，顾客需要花费大量的时间才能找到自己想买的商品。这种浏览大量无关的信息和产品过程无疑会使淹没在信息过载问题中的消费者不断流失。为了解决这些问题，个性化推荐系统应运而生。

推荐算法在互联网行业的应用非常广泛，今日头条、美团点评等都有个性化推荐，推荐算法抽象来讲，是一种对于内容满意度的拟合函数，涉及到用户特征和内容特征，作为模型训练所需维度的两大来源，而点击率，页面停留时间，评论或下单等都可以作为一个量化的 Y 值，这样就可以进行特征工程，构建出一个数据集，然后选择一个合适的监督学习算法进行训练，的都模型后，为客户推荐偏好的内容，如头条，就是咨询和文章。

产生于电子商务领域的个性化推荐技术是基于对用户的偏好进行分析对用户推荐商品的一种创新性的方法，这种方法会大大提高商品成交的纪律，如今个性化推荐的算法

已经越来越完善，而且效果越来越好，如果我们将其应用到新闻媒体行业，是否会为这个行业注入新鲜的活力呢？文本通过介绍小组设计过程并结合现存的新闻推荐程序，学习实现个性化推荐算法。

1.2 背景

随着万维网的发展，人们的新闻越度习惯逐渐从传统的报纸、电视等媒介转移到互联网上。在线新闻网站，如 Google News 和 Bing News，从各种来源收集新闻，为读者提供新闻的总览。网络新闻平台的一个最大的问题是，文章的数量可能会让用户望而却步。各类新闻平台都存在信息过载的问题，为了减轻信息过载的影响，关键是要帮助用户确定他们的阅读兴趣，并提出个性化推荐。

互联网在最近十年内出现了爆发性增长，伴随着用户数量迅速的增长的背后是用户数据量的指数级增长，面对着海量的信息，用户往往后感到束手无策，这就是互联网中所谓的信息过载问题。如何帮助用户从海量的信息中获取用户最感兴趣的信息逐渐成为一项热门的研究工作。

信息过载问题传统的解决方案是基于被动响应的服务模式，即用户提出有针对性的需求，服务器端则根据每个用户的具体需求，过滤用户不感兴趣的一些信息，然而这种解决方案也存在一些缺陷，因为它只是向用户提供一些共同兴趣点比较高的信息，例如热门的新闻、电影、音乐等，无法满足用户日益增强的个性化需求，用户获取个性化信息的难度仍然很大。

个性化的解决方案是解决上述问题的主要方法，通过分析用户的历史数据对用户的兴趣爱好进行建模，为每个用户创建一个 profile 文件，其中记录用户的兴趣表示，并能在和用户不断的交互中学习用户的兴趣，即使更新用户的 profile，在适当的时候提供给

用户其感兴趣的信息。

近年来个性化信息服务逐渐成为 Web 技术的热点，推荐系统在实时资讯、新闻、微博、电影、音乐、博客、电商等 Web 站点中都有大量的应用。通过推荐系统，系统可以有效地解决信息过载问题，分析用户的评分与行为等历史数据建立用户兴趣模型，无需用户特意地填写大量的兴趣调查信息，极大的减轻了用户的负担，使用户的认可度大大增加。

另一方面海量数据的运营压力对每个公司也是非常巨大的，每天都可能会有大量的物品上架，下架。常用的推荐算法一般都基于用户评分历史数据，新上架的物品由于没有任何的访问记录，因此很难将其推荐给对其感兴趣的用户，这种物品有可能一上架后就面临这没有机会被访问的问题，这就是推荐系统中常提到的冷启动问题，因此配合推荐系统的物品运营系统也是非常重要的一个部分，运营人员可以通过人工归类、打标签、写商品描述的方式，为新增的物品提供一些初始的信息，这样物品可以迅速被推荐引擎挖掘，并推荐给感兴趣的用户。

1.3 系统发展历程

1995 年 3 月，卡耐基·梅隆大学的 Robert Armstrong 等人在美国人工智能协会上提出了个性化导航系统 Web Watcher；斯坦福大学的 Marko balabanovic 等人在同一会议上推出了个性化推荐系统 LIRA

1995 年 8 月，麻省理工学院的 Henry Lieberman 在国际人工智能联合大会（IJCAI）上提出了个性化导航智能体 Litizia；

1996 年，Yahoo 推出了个性化入口 My Yahoo；

1997 年，AT&T 实验室提出了基于协同过滤的个性化推荐系统 PHOAKS 和 Referral

Web；

1999 年，德国 Dresden 技术大学的 Tanja Joerding 实现了个性化电子商务原型系统 TELLIM；

2000 年，NEC 研究院的 Kurt 等人为搜索引擎 CiteSeer 增加了个性化推荐功能；

2001 年，纽约大学的 Gediminas Adoavicius 和 Alexander Tuzhilin 实现了个性化电子商务网站的用户建模系统 1：1Pro；

2001 年，IBM 公司在其电子商务平台 Websphere 中增加了个性化功能，以便商家开发个性化电子商务网站；

2003 年，Google 开创了 AdWards 盈利模型，通过用户搜索的关键词来提供相关的广告。AdWords 的点击率很高，是 Google 广告收入的主要来源。2007 年 3 月开始，Google 为 AdWords 添加了个性化元素。不仅仅关注单词搜素的关键词，而是对用户近期的搜索历史进行记录和分析，据此了解用户的喜好和需求，更为请确地呈现相关的广告内容。

2007 年，雅虎推出了 SmartAds 广告方案。雅虎掌握了海量的用户信息，如用户的性别、年龄、收入水平、地理位置以及生活方式等，再加上对用户搜索、浏览行为的记录，使得雅虎可以为用户呈现个性化的横幅广告。

2009 年，Overstock（美国著名的网上零售商）开始运用 ChoiceStream 公司制作的个性化横幅广告方案，在一些高流量的网站上投放官品广告。Overstock 在运行这项个性化横幅广告的夫妻就取得了惊人的成果，公司称：“广告的点击率是以前的两倍，伴随而来的销售增长也高达 20%至 30%”

2009 年 7 月,国内首个个性化推荐系统科研团队北京百分点信息科技有限公司成立，该团队专注于个性化推荐、推荐引擎技术与解决方案，在其个性化推荐引擎技术与数据平台上汇集了国内外百余家知名电子商务网站与资讯类网站,并通过这些 B2C 网站每天

为数以千万计的额消费者提供实时智能的商品推荐。

2011 年 8 月,纽约大学个性化推荐系统推荐团队在杭州成立载言网络科技有限公司，在传统协同滤波推荐引擎基础上加入用户社交信息和用户的隐性反馈信息，包括网页停留时间、产品页浏览次数，鼠标滑动，链接点击等行为，辅助推荐，提出了迄今为止最为精准的基于社交网络的推荐算法。团队目前专注于电商领域个性化推按服务以及商品推荐服务社区——e 推荐。

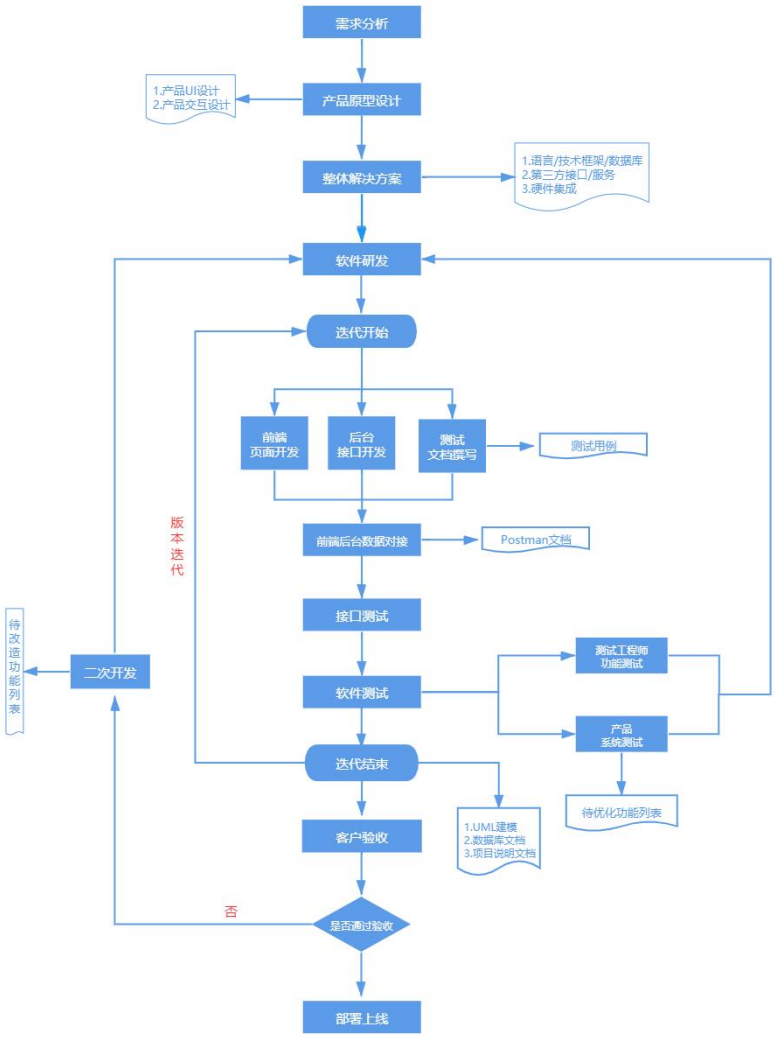
2011 年 9 月，百度世界大会 2011 上，李彦宏讲推荐引擎与云计算、搜索引擎并列为未来互联网重要战略规划与发展方向。百度新首页将逐步实现个性化，智能地推荐出用户喜欢的网站和经常使用的 APP。

1.4 课程设计目的

本次课程设计主题为“新闻个性化采集推荐系统”，小组成员拟通过爬虫爬取页面、提取正文内容、文本分析、文本相似度计算、机器学习分类算法、搭建网络页面、网络前后端连接、数据库连接等技术，合作完成一个个性化采集推荐系统。

程序开发流程仍沿用软件开发的思想：可行性研究与构思-->需求分析-->设计-->开发-->测试-->维护。

通过此次课程学习，计划小组成员按分工可分别熟悉和掌握前端、后端和算法三部分技术和流程，并清晰地理解关键的对接问题，为以后在后续学习和工作中积累宝贵的经验和技术。



我们选择了“新闻个性化采集推荐系统”作为本课程的选题。小组各成员通过学习前端设计、爬虫构造、数据库连接、文本分析、深度学习算法等，合作实现这样一个个性化采集推荐系统。同时整个软件设计的流程，按照软件工程所学的步骤有条不紊进行，从而巩固所学、培养良好习惯。长达七天的分工合作来实现一个项目，让各成员面对、适应将来在实践、工作中可能会遇到的各类问题。

本次实践主要选用的开发平台为 Pycharm，主要选用的编程语言为 Python，选用的数据库为 Mysql。

二、需求分析

推荐系统越来越成为互联网中不可或缺的一部分，一个成熟的推荐系统中一般由都会有用户与运营专员两个角色，用户通过在推荐系统中注册账户来在系统中唯一的标识自己，新的用户注册完毕后需要填写一些初始化的兴趣信息，以帮助系统解决用户没有任何历史数据记录而产生的冷启动问题，用户可以通过点击感兴趣的新闻与推荐系统产生交互，推荐系统会储存用户的历史操作数据，并利用数据分析引擎对历史数据分析建模为用户生成推荐列表；另一个比较重要的角色就是运营人员，运营人员负责更新新闻数据集，每天都有大量的新闻发行，运营人员需要收集相关的新闻资料，将其输入进系统中，并填写一定的标签，以方便推荐引擎即时的发掘新的电影并将其推荐给感兴趣的用户。

2.1 功能需求

2.1.1 用户功能需求

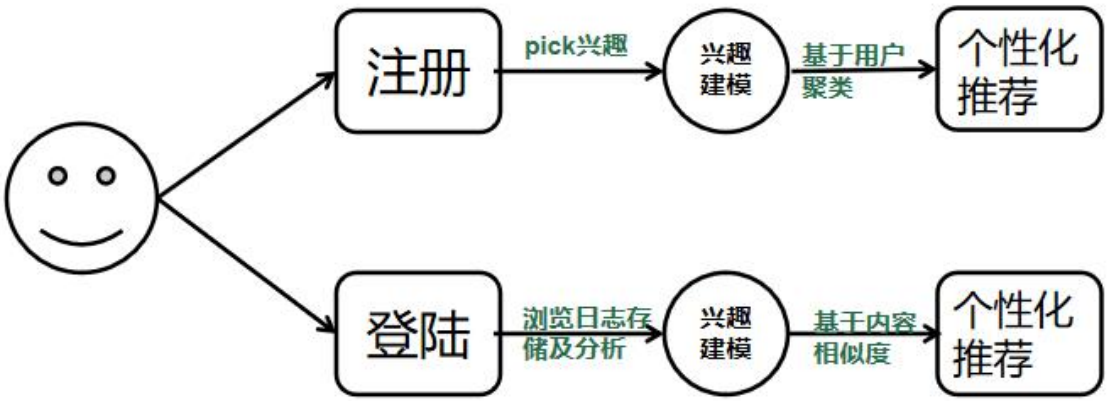


图 2-1，用户功能需求模型

系统提供给用户的初始界面是注册登陆界面，用户按照流程首次注册之后，用户的 username 和 password 便会通过与后端相连接的 mysql 进行存储，登录成功之后，会由用户首先挑选感兴趣的话题，我们小组提供了十四个类型新闻，包括：财经 彩票 房产 股票 家居 教育 科技 社会 时尚 时政 体育 星座 游戏 娱乐。基于用户挑选的类型，计算新用户与已存用户的聚类相似度，基于高相似用户之间的兴趣相同点对新用户推荐首次页面。之后用户对于新闻的浏览记录存储在日志并返回给后端，后端存储与 mysql 数据库中。算法设计基于 mysql 存储的浏览记录对用户进行动态建模，并随着用户刷新页面动态返回用户的兴趣最大化。在之后的登陆时，就可以直接基于用户唯一的兴趣模型进行个性化推荐，并且也会动态的记录用户的浏览日志并进行分析，解决用户的兴趣随时间变化而转移的问题。

名称	用户注册表
功能描述	用户首次注册时相关信息，包括用户名，邮箱，密码等信息
优先级	必要
输入	用户的基本信息
前端页面	1
输出	将用户填写的内容存储于数据库
补充说明	首次注册的用户名不可以与已有的用户名相同

表 2-1，用户注册表

名称	用户登陆表
功能描述	用户登陆时，提供 username 和 password，查询数据库中的数据
优先级	必要
输入	用户的 username 和 passwprd
前端页面	1
输出	将用户填写的内容和数据库中的数据进行比较
补充说明	用户提供的 username 和 password 必须和数据库存储的信息相匹配

表 2-2，用户登陆表

名称	用户挑选兴趣表
功能描述	用户对呈现的新闻分类挑选感兴趣的类型
优先级	重要
输入	用户挑选的类型
前端页面	2
输出	将用户挑选的类型存储到匹配的数据库
补充说明	用户首次挑选的兴趣类型将用于对用户的建模分析，呈现首个页面

表 2-3，用户挑选兴趣表

2.1.2，运营功能需求

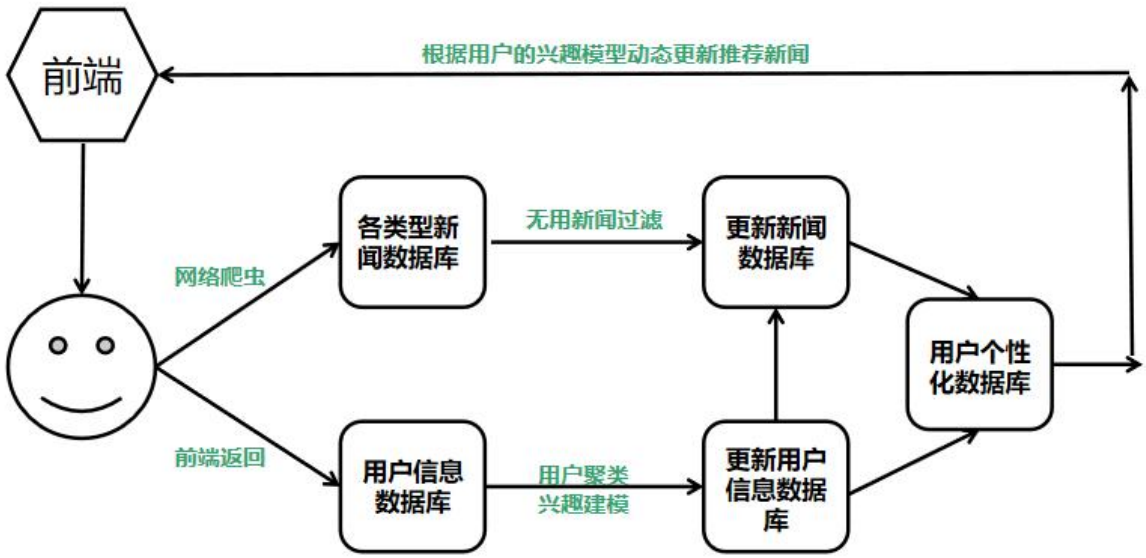


图 2-2，运营功能需求模型

后端运营者首先通过网络爬虫爬取各个类型的新闻数据，然后构建一个新闻数据库。

后端接受用户用于注册登陆的信息，构建用户信息库，在用户之后登陆时，便可以直接遍历数据库进行比对，就可以检查用户信息的正确性。在用户浏览一部分日志之后，前端将日志信息返回给后端，后端运营机器学习的算法，重新构建用户的兴趣模型，然后再有过滤性的爬取各个类型的新闻，构建每个用户的个性化新文库。之后返回给前端，用于实时生成个性化推荐页面。

名称	前后端交互表
功能描述	接受前端返回的用户信息，并于数据库进行交互
优先级	重要
输入	前端输入的信息
输出	将前端传递的信息储存在数据库中
补充说明	前后端的交互是密切的，需要持续交互

表 2-4，前后端交互表

名称	后端爬虫表
功能描述	在网络上爬取各个新闻类型的文章
优先级	重要
输入	爬取新闻的链接
输出	爬取到的新闻及 url

补充说明	爬虫第一次爬取各个类型的文章，之后按照用户兴趣模型过滤的爬取一些文章

表 2-5，后端爬虫表

名称	后端过滤表
功能描述	根据用户的兴趣模型过滤新闻库中无用的新闻信息
优先级	重要
输入	用户的兴趣模型
输出	基于用户兴趣的新闻及 url
补充说明	由于用户的兴趣模型随着浏览日志是一直动态变化的,所以后端过滤模型需要及时更新

表 2-6，后端过滤表

名称	后端更新表
功能描述	根据用户的兴趣模型和爬取的个性化新闻构建个性化新闻库
优先级	重要
输入	用户的兴趣模型和爬取的新闻
输出	个性化的新闻库

补充说明	对于用户的兴趣需要与用户一一匹配

表 2-7，后端更新表

2.1.3，算法功能需求

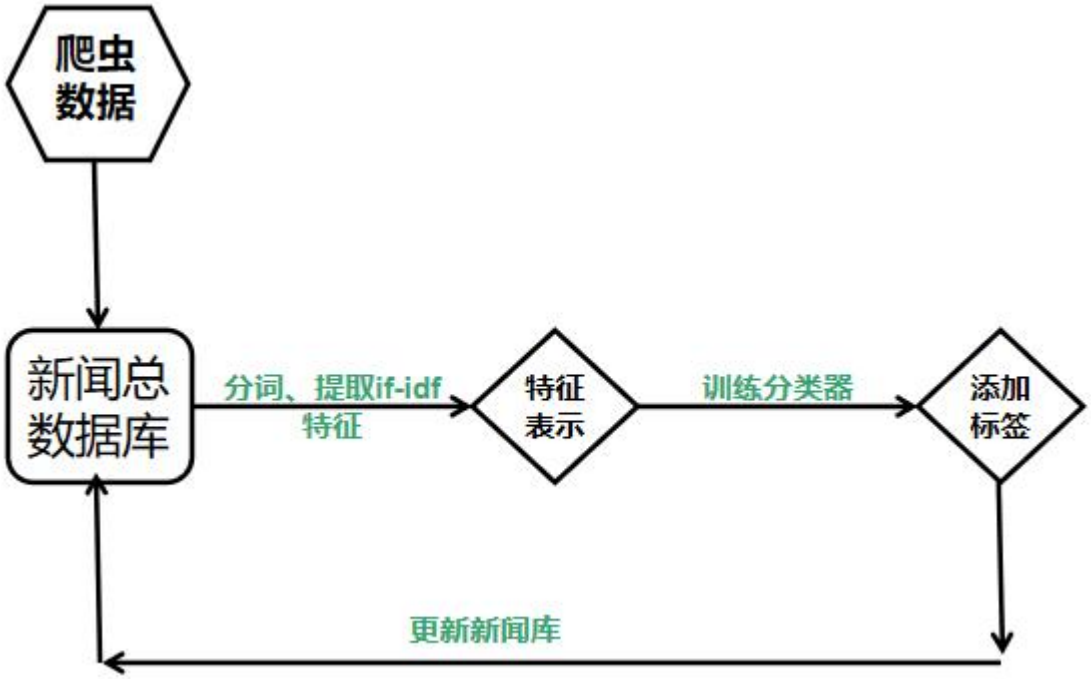


图 3-1，新闻数据库功能需求分析

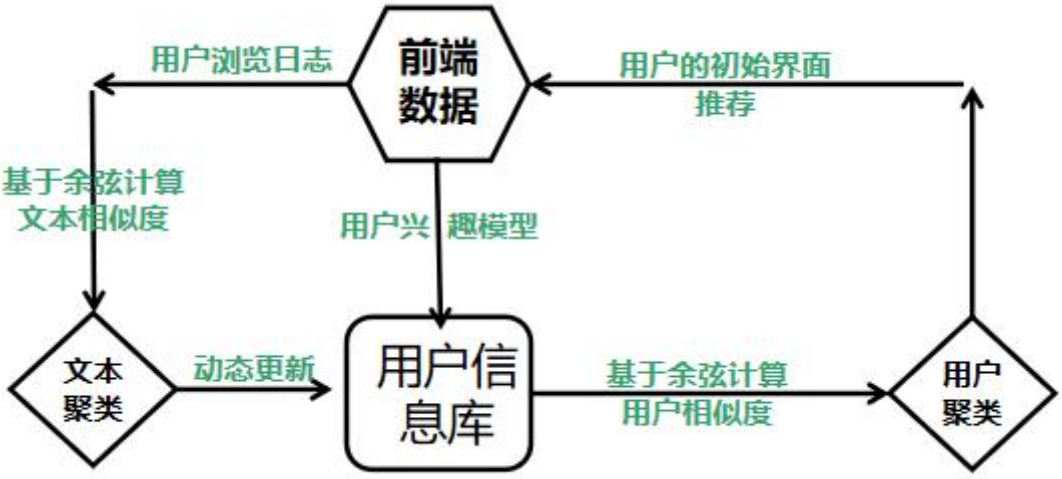


图 3-2，用户信息库功能需求分析

后端首先从网络平台上爬取无标签的各种新闻，之后经过分词、特征提取等使用训练好的分类器对新闻进行分类，并且给每个新闻添加一个类别标签。然后用于重新更新新闻库。由于首次登陆页面的用户没有浏览记录，唯一获取兴趣点的便是首次挑选的兴趣标签，通过余弦计算该用户与已有用户的相似度，相似度大于某一阈值的用户，认为新用户与该类用户属于同一类别，然后基于该团体的兴趣标签生成新用户的首次推荐新闻页面。在用户浏览过一定量的新闻之后，通过前端的日志记录，返回给后端之后，后端给其兴趣重新建模分析，之后基于文本内容计算内容的相似度，给用户推荐的页面就是基于文本内容的相似重新推荐新闻。

名称	后端分类器
功能描述	根据用户的兴趣模型对爬取的新闻文本进行分类
优先级	重要
输入	用户的兴趣模型
输出	对所爬取的新闻文章加上分类后的标签
补充说明	当文本的数量足够多时，才能达到较高的准确率

表 2-8，后端分类器表

名称	后端聚类表
功能描述	对于相似用户进行聚类，对于相似文本进行聚类
优先级	重要
输入	新用户的兴趣模型
输出	相似文本的输出
补充说明	新用户首次注册时，基于用户聚类；分析文章日志是基于文本相似度

表 2-9，后端聚类表

2.2，非功能需求

2.2.1 性能需求

1）内存的容量

由于系统中涉及到大量的用户，新闻文本相似度计算，因此系统的内存容量需要尽可能的满足存储系统计算的中间数据，如用户相似度矩阵，文本相似度矩阵，用户的浏览日志列表，等信息。

2）外存的容量

外存的容量也相当重要，大部分的浏览记录与文本信息都需要存在数据库中，系统外存需要有足够的空间满足这些存储要求。

3）查询用户推荐列表速度

用户点击网页，浏览新闻的速度比较快，系统需要在短暂的时间内计算出推荐列表并返回给用户浏览器，让用户及时地看到推荐给其的列表记录，这样用户才会对推荐的列表进行反馈，并进行后续的与推荐系统的交互。

2.2.2 准确性需求

推荐系统推荐给用户的电影列表需要有足够的准确率,这样才能增强用户的认同感，一个好的推荐系统，准确性一定是一个重要的指标。

2.2.3 稳定性需求

推荐系统的各种算法都有其各自的优缺点，例如，过滤算法与基于用户相似度的冷启动问题与数据稀疏性问题，当浏览记录不足给用户准确的推荐，而基于内容过滤的算法则不存在这一问题，另一方面，在相似度计算问题上，有的推荐是有对称性的，例如喜欢《数据挖掘概念与技术》这本书的人可能会喜欢《数据挖掘原理》，而喜欢《数据挖掘原理》也喜欢《数据挖掘概念与技术》，但是在购物篮问题中，买了装修地板的人很有可能也去买装修的电器，而买电器的人不一定会去买装修地板，对于前者使用聚类相似度算法比较合适，因为它是对称的，而后者使用文本相似度算法比较合适。综上所述针对具体情况选择不同的推荐算法很重要，这对大大提高推荐系统的推荐准确度非常有帮助，还能降低系统模块之间的耦合度。

2.2.5 可靠性需求

推荐系统的运营模块是一个非常重要的环节，数据不一致导致的问题很有可能使得

用户无法正常地使用推荐系统，另外，大量的数据需要运营，新的新闻都需要运营人员迅速的加入推荐系统中，利用运营专员提供的初始的新闻信息，将这些新闻文本推荐给可能感兴趣的用户，用来提高推荐系统的竞争力，特别是在目前的移动网站的建设中，可靠性的要求非常高，对各种移动平台的支持，需要特定大小的图片，特定布局的网页设计才能使得各种平台上的用户使用起来不出问题。

2.3 本章小结

本节对系统的功能性需求和非功能性需求进行了细致的分析，包括了用户的登陆，用户注册，用户浏览日志，运营人员的新闻信息推荐，后端算法的原理等功能性需求，另外系统的性能，准确性，可靠性与可扩展性这些非功能性需求也进行了一一介绍。

三、详细设计

3.1. 系统结构设计

本系统的架构设计如下图 3-1 所示：

前端层主要是负责前端页面展示，使用 html，CSS，JavaScript 等技术实现，用户可以挑选喜欢的新闻类型，也可以随便浏览喜欢的新闻。前端还有相关业务的服务和个性化推荐服务。

交互层主要是负责前端和后端的数据交互，前端传给后端的信息包括用户用于注册登陆的 username 和 password，并实时记录用户的浏览日志，待用户刷新页面时发送给后端相关文件。后端发送给前端的文件主要为根据用户的反馈计算出的相关新闻推荐。

Common Service 提供用户获取用户推荐新闻列表，获取根据新闻推荐相关新闻列表的功能。由于 Common Service 使用频率比较高，因而使用 python 实现并对外提供 python 调用接口，供 Java 调用。Analysis Service 是一个定时触发的线程，该线程定期分析用户的日志数据获取用户的浏览记录，更新用户数据库，重新计算用户与用户，新闻文本与文本之间的相似度，更新用户与新闻的 profile 信息。

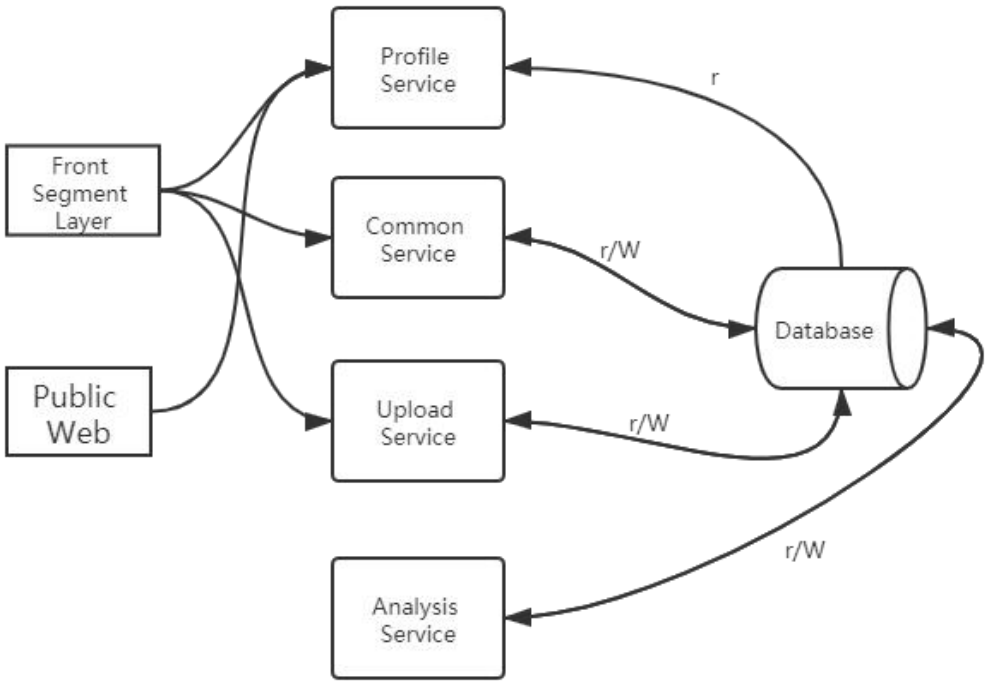


图 3-1，系统架构图

3.2. 模块设计分析

3.2.1 系统功能

新闻个性化推荐系统模块的功能分为前端层、交互层、后端层、算法层、数据库、运行环境等六个层次。前端层提供展现给用户的页面设计；交互层负责前端和后端的信息传输；后断层实现存储前端返回的信息，并把相关的信息存储于数据库中；算法层包括用户聚类算法和文本聚类算法，并根据具体情况采用不同的推荐算法；数据库存储用户的基本信息以及用户浏览日志数据；运行环境包括 python、java、html 等。

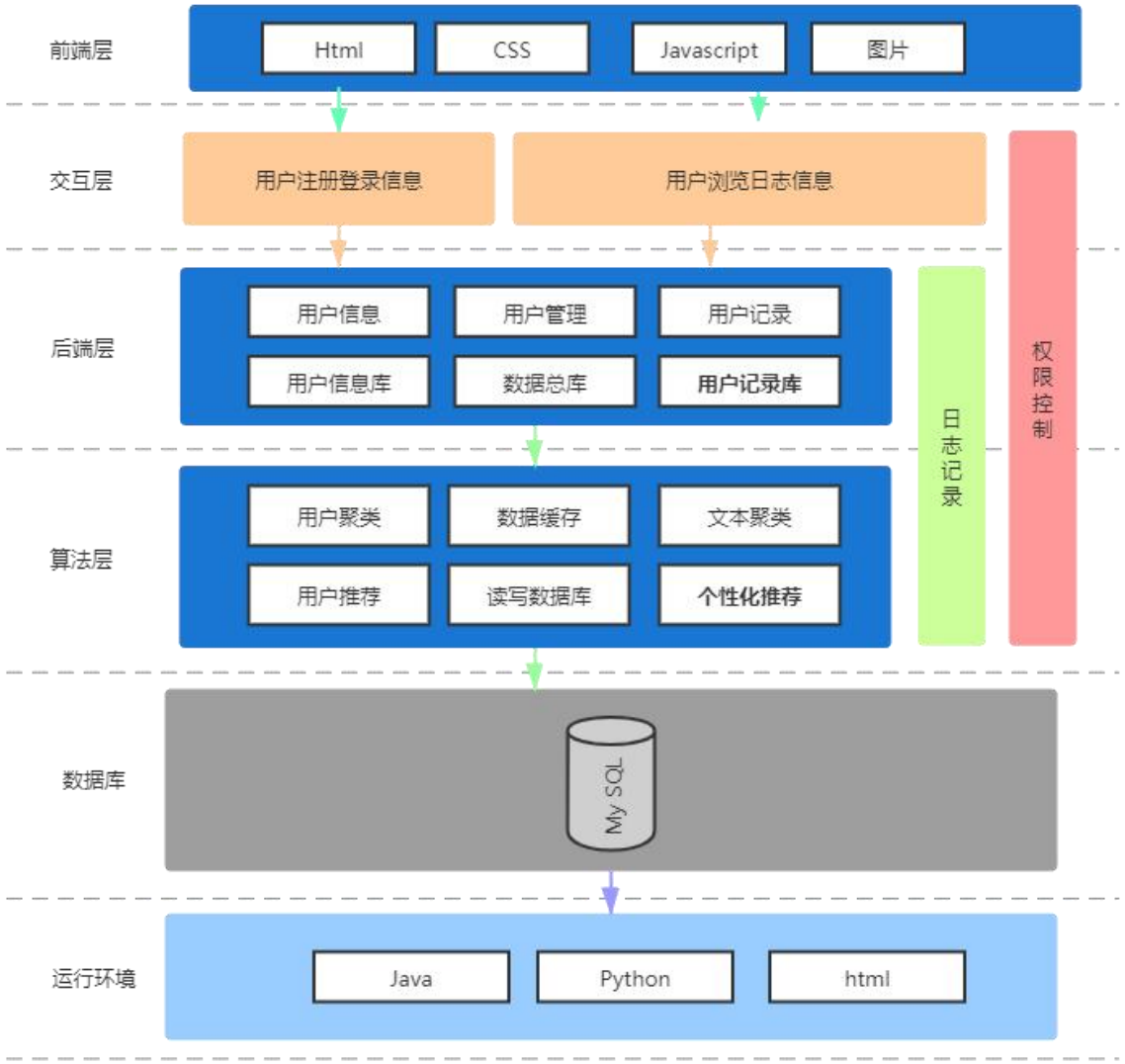


图 3-2，系统功能模块图

- 用户信息管理

推荐系统中新用户注册后会填写注册信息与一些个人兴趣爱好标签信息，随后该用户可以进行正常的浏览等其他的操作，随着系统与用户的交互，系统逐渐获取用户的兴趣的兴趣爱好。用户可以对自己的标签信息进行管理，通过对自己喜爱的新闻类型多次浏览等操作，系统会将用户的浏览行为存入日志，并调用数据分析引擎对用户的行为进行分析，更新用户的 profile 文件。

- 新闻信息运营管理

推荐系统日常的运营是由运营人员负责的，运营人员每天会将新的新闻爬取到数据库中，并根据分类器实现贴标签，以方便推荐系统更好的解决冷启动问题，将这些还未被浏览的新闻更够有机会准确的推荐给感兴趣的用户

- 推荐引擎

推荐引擎时推荐系统中最核心的一部分，通过对用户挑选的兴趣和用户每天的浏览记录，利用余弦聚类算法计算出和用户有较高相似度的其他用户，并结合二者相同的兴趣标签，可以更准确的推荐所喜欢的标签。对于新注册的用户，推荐系统会强制用户选择一些其感兴趣的标签和新闻分类，随后推荐引擎会根据用户提供的这些标签信息，获取属于该标签的一些热点的新闻，作为用户初次浏览的页面。

- 数据分析引擎

数据分析引擎对用户的操作日志进行分析，获取用户的操作记录并将该信息爬取为推荐引擎的原始数据来源，对数据进行处理之后，存入数据库中。当用户的兴趣发生变化时，这些将会反映在用户的浏览记录中，数据分析引擎会捕捉到，并重新计算用户的 profile，并更新用户和相关新闻的 profile 文件，这样推荐引擎将会根据新的 profile 文件为用户生成新的推荐列表。

3.2.2 系统服务部署架构

个性化新闻推荐系统的基本架构如下图 3-3 所示：

用户通过智能手机、移动终端、或者 PC 登陆系统，Web 服务器会将用户的注册信息和用户的操作记录提高给日志服务器，日志服务器中记录的日志主要有用户的浏览信息：用户浏览过的页面、用户浏览页面的时间等。数据分析服务器用于从日志服务器中获取一段时间的日志文件，对这些日志文件进行分析重新计算用户之间的相似度，新闻

文本之间的相似度并更新相关用户的 profile 文件。推荐引擎在用户挑选兴趣标签、浏览页面时触发，Web 服务器获取用户浏览页面的信息，将这些信息提交给推荐引擎，推荐引擎根据用户和新闻文本的信息，同时从数据分析服务器中获取相应的相似度信息和用户 profile 文件，根据相应的推荐算法计算推荐列表给用户推荐。

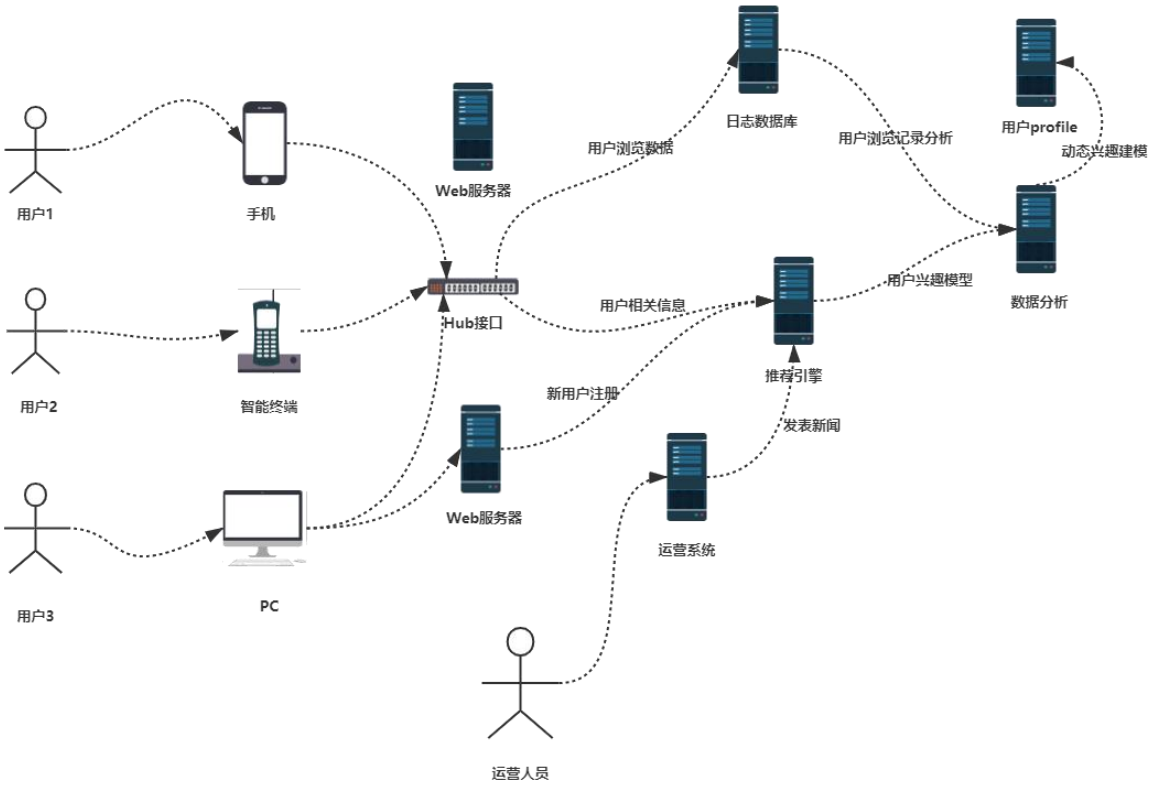


图 3-3 推荐系统运行模式图

3.2.3 数据时序图

时序图是一种图形化的系统模型，它在一张图中展示信息系统的主要要求，即输入、输出、处理（过程），数据存储，如下图 3-4。

1）新用户注册

新用户注册后，会建立新的用户 profile 文件，根据用户填写的兴趣标签更新用户的相似

度列表与用户的推荐列表。

2）新的新闻发表

新闻发表后，推荐系统会添加新的新闻标签，根据运营人员填写的新闻的分类更新相应的新闻文本相似度数据，计算相应的感兴趣的用户类表，更新这些用户的推荐列表信息，并在这些用户登陆系统后给其推荐。

3）用户挑选兴趣

用户在初次登陆时，会强制用户挑选感兴趣的标签，并进行提交之后会实时存储在用户的数据库中，后端算法读取数据库中所选择的兴趣标签会对用户进行建模，方便算法基于用户聚类的特点对新注册用户个性化推荐新闻。

4）用户浏览新闻网页

用户的兴趣兴趣反映在用户对新闻的溃烂行为，当用户从推荐列表中选取了感兴趣的电影后，相关数据就会存储在数据库中，并发送到数据库中，方便后端算法对数据进行计算，对用户的兴趣模型进行重新建模。

5）用户注销帐号

当用户对该网站不再满足时，选择注销帐号时，提交相关请求后，前端会将请求发送给后端，后端会及时从数据库中删除用户的相关信息，数据库中不再存在用户的全部信息，用户之后采用相同的信息进行注册时，在后端数据库中可以及时存储用户的信息，并建立用户的兴趣模型，与之前的数据模型不会有任何的关系，对用户重新建模，重新进行新闻文本推荐。

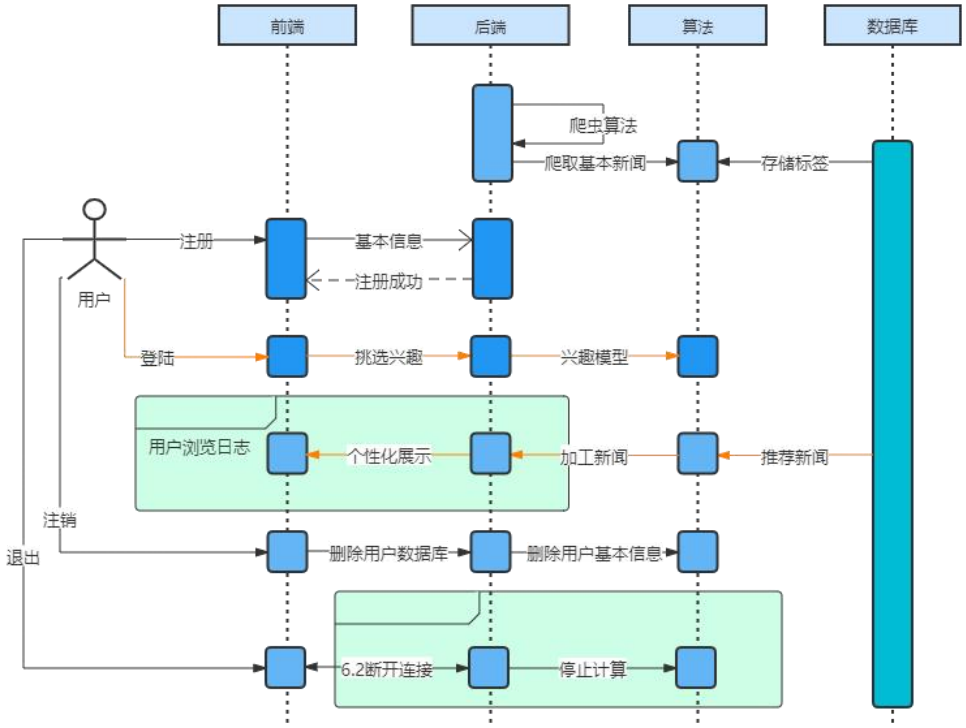


图 3-4，用户运行时序图

3.3. 重要数据结构设计

3.3.1 新用户注册

新用户注册时，用户通过 Web 页面向 Web 服务器提交用户的注册信息，这其中包括用户的基本信息和用户的初始兴趣标签，Web 服务器会将这些信息提交给推荐系统的注册管理模块，注册管理模块提取出用户的基本信息存入数据库中，将标签信息等有助于获得用户兴趣的信息提交给推荐引擎，推荐引擎查询数据库，获取具有相似标签的热门新闻和比较热点的新闻作为推荐列表返回给用户 Web 页面，这样用户在注册完毕后即可享受到推荐系统给其推荐的电影。用户的初始兴趣模型即为选取的初始标签，后期的兴趣模型与用户浏览的日志文件相关，对用户进行实时的兴趣建模，尽量提高用户兴趣的准确性。

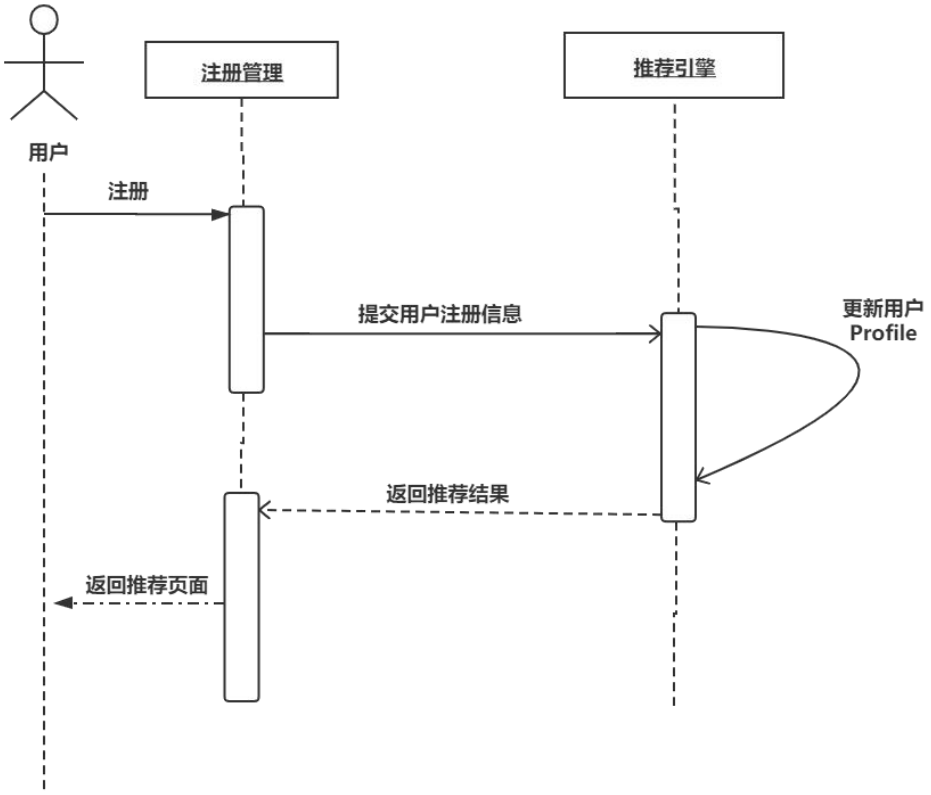


图 3-5 新用户注册图

3.3.2 新的新闻文本发表

推荐系统一般都有一个专职的营运专员，负责推荐系统的日常运营工作，包括爬取新的新闻文本，完善爬取文本的信息，因而推荐系统有一个很重要的分支模块就是后台运营系统。当一个新的新闻发表时，运营专员将新闻的基本信息通过后台运营系统提交，这些信息包括新闻的标题、发表日期、出处、连接、作者等。后台运营系统将相应的新闻文本信息存入数据库中，接着会向推荐引擎提交新发表的新闻的基本 i 西南西，推荐引擎将会按照特定算法给文本进行分类，添加标签信息，运营人员根据分类的标签提交给推荐引擎，推荐引擎根据新的文本标签，重新计算文本的相似度信息和 profile 信息，更新数据库中的记录。并获取对该文本感兴趣的用户列表，将该新闻文本加入到这些用户的推荐列表中。

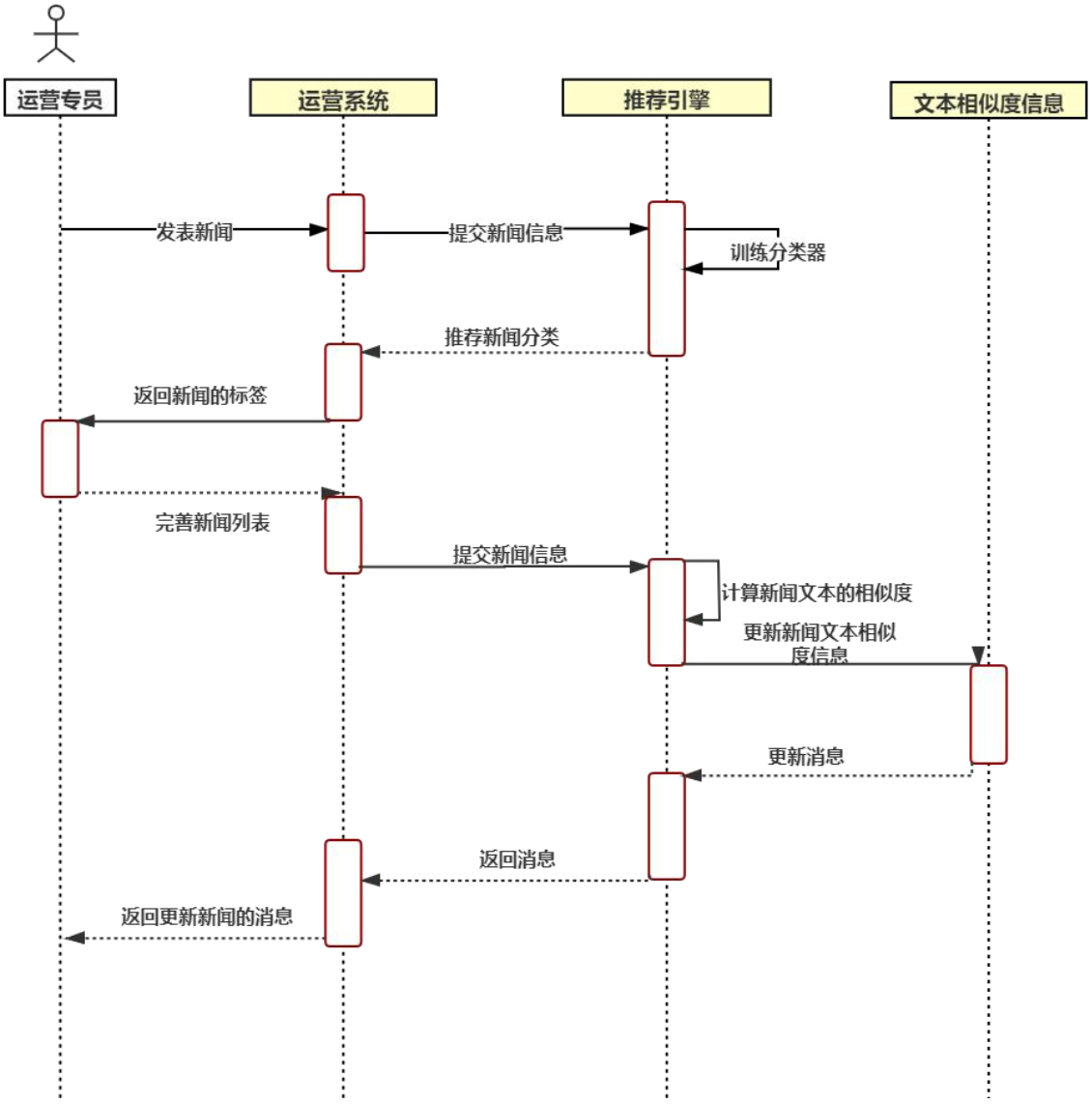


图 3-6 新的新闻发表

3.2.3 给用户推荐新闻文本

推荐系统中使用的最为频繁的功能就是获取用户的推荐列表这一功能，在这个过程中，用户首先会通过浏览器向 Web 服务器发送一个获取推荐列表的请求，Web 服务器会提取出这个请求中的用户新闻信息将其转发给推荐系统的应用服务，应用服务接着调用推荐引擎的获取推荐列表功能，推荐引擎收到调用请求后查询到相应的用户的兴趣模型和电影的标签，利用推荐算法计算推荐列表，当用户量大时，推荐系统会维护一个用

户的推荐列表数据库，直接从数据库中读取推荐列表信息将其推荐给用户，以避免耗时的计算过程，将耗时的计算过程定期在用户每次提交个性化申请时。

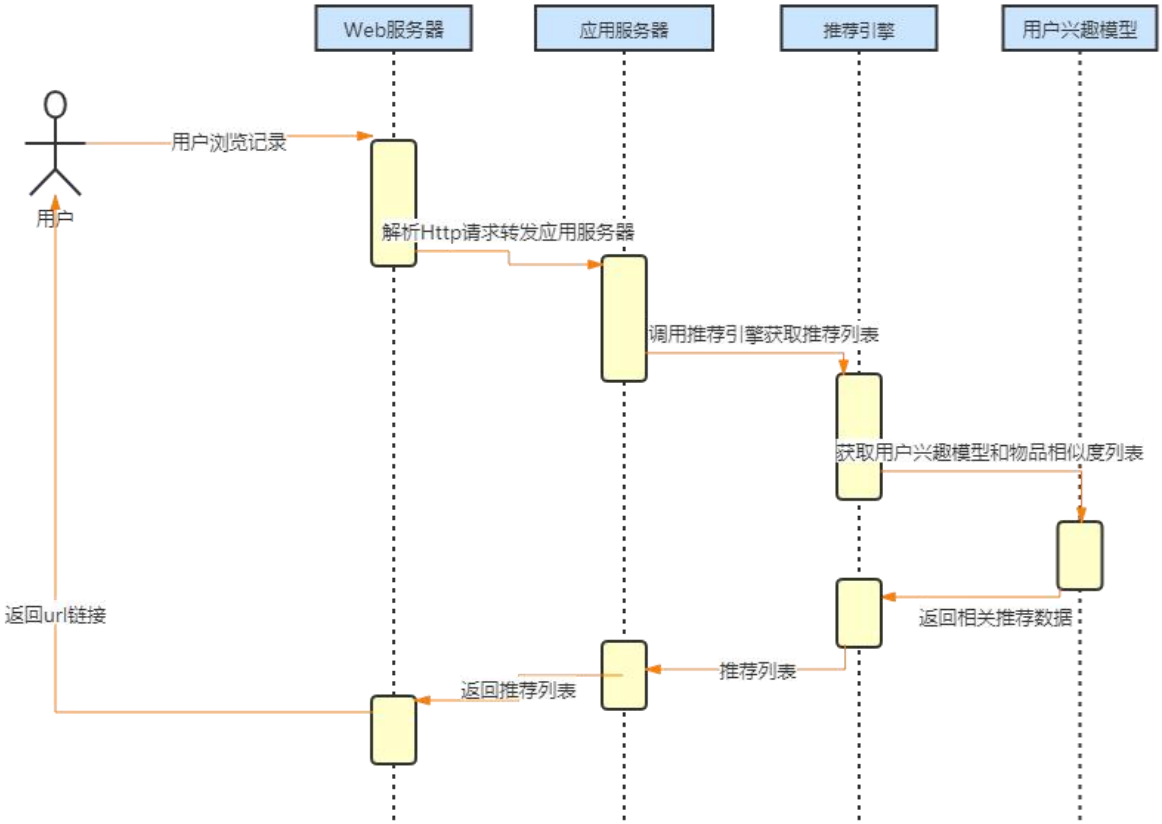


图 3-7 推荐数据

3.2.4 后台数据分析

后台数据分析这个过程主要是用于在用户访问时，对用户的浏览日志数据进行分析整理，调用分类器算法对新的文本进行分类，调用聚类算法，计算用户的兴趣模型，用户刷新页面时，数据提交会被触发，向后端发送浏览日志记录，后端会同步读取浏览的相关的网页内容并找到内容相似度较高的相关文本，并返回给前端相关的推荐数据，用于生成用户的刷新页面。

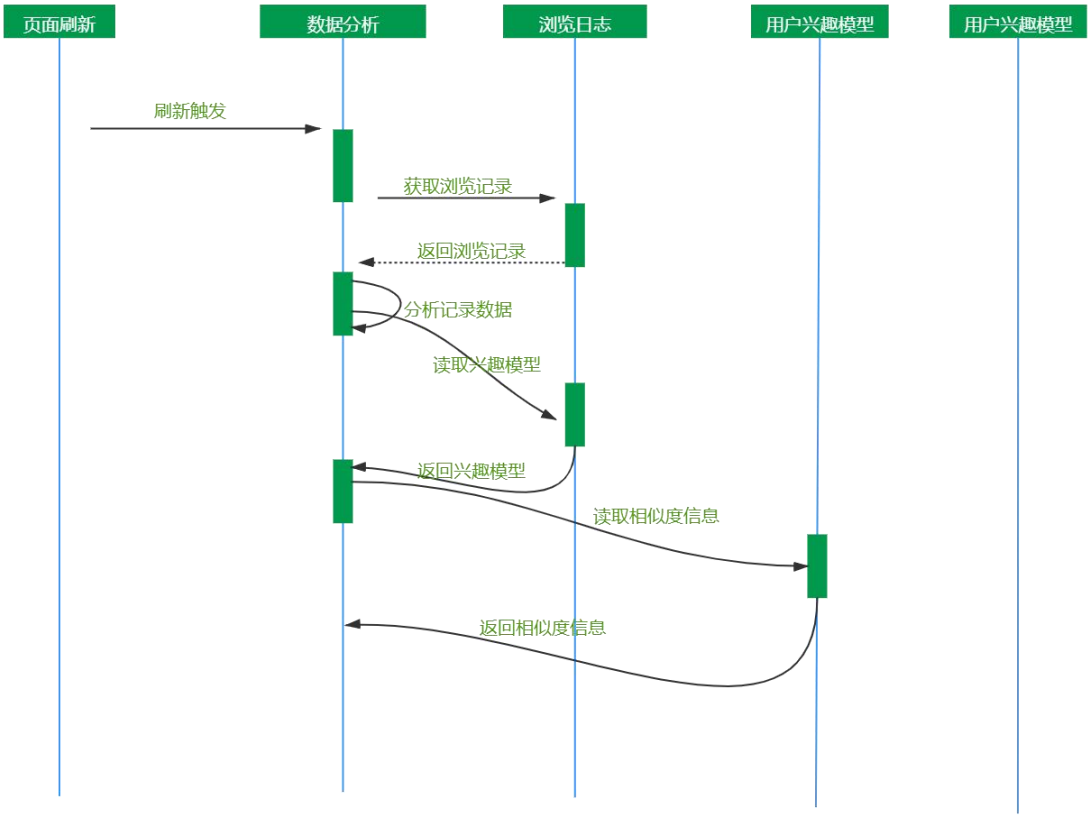


图 3-8 后台数据分析

3.4. 程序函数清单

3.4.1 后端部分函数清单

1）函数名——所在文件名 Java类：

Logininervelet

RegisterServelet

ClickServelet

CheckboxServelet

infoServelet

getRequestDispatcher

setCharacterEncoding

getParameters

ifTableNull

2）函数功能

RegisterServelet：注册功能的实现

Logininervelet：登录功能的实现

CheckboxServelet：兴趣爱好多选并返回到后台

InforServelet：将内容展示到前端，即将信息反馈到html页面让用户能够看到

ClickServelet：记录用户是否点击过当前页面

getRequestDispatcher:实现页面的跳转

sendRedirect：实现页面的跳转

setCharacterEncoding：编码格式定义（避免乱码，一般是UTF-8）

WriteSql（我们定义的一个类）类中的函数：

Connection：与数据库的连接

Register：sql语句解决注册过程中在数据库中创建新的元素。

Seek：sql查询登录过程中的ID和密码

Seekuser：注册过程查询数据库中的ID（就是为了确认是否现在输入的ID数据库中已经存在了）

GetPrameters:用于读取提交表单中的值，即获取POST/GET传递的参数值。

ifTableNull：判断表中14个hobby是否全为空（全为0）

3）参数说明

由于参数传递较多，我们主要介绍关键的参数。

RegisterServelet：regid注册名字 regpassword1 regpassword2分别是两次输入的密码，需要两次输入的密码都相同才能注册成功。

Register：id password

Logininervelet：id password 名字和密码

Seek：id password bool型 返回 true 和false

seekuser：id password bool型 返回 true 和false

getPrameter： string name name要获取参数的名称，返回值指定名称的参数

ifTableNull：输入id 返回true 或者 false bool

4）算法描述

RegisterServelet：通过改写的 doGet doPost 实现基本功能在doPost中提前预设好编码格式UTF-8，否则会乱码。通过几个if else 语句实现基本的逻辑判断。大题思路可由如下图所示

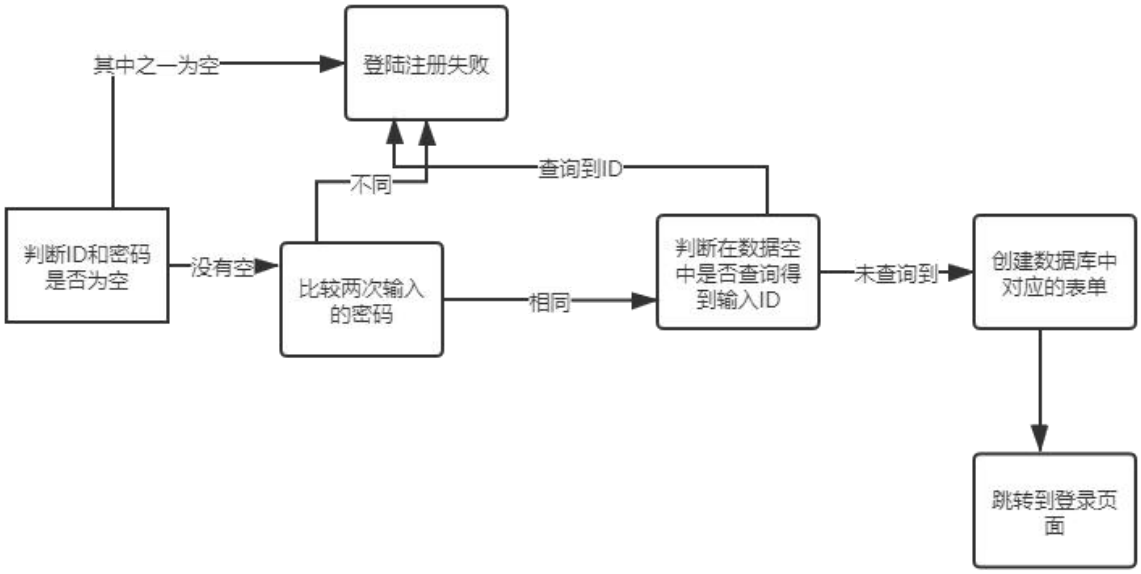


图3-5，后端运行流程图

最后的跳转通过sendRedirect实现跳转到登录页面。

备注：可以使用sendRedirect和getRequestDispatcher实现页面跳转，但是我们在实现的

过程中发现getRequestDispatcher会出现不可避免的乱码，因此我们选择在这部分使用了sendRediect。GetRequestDispatcher是服务器内部跳转，地址栏信息不变，只能跳转到web应用内的网页。SendRedirect是页面重定向，地址栏信息改变，可以跳转到任意网页。

Logininervelet：判断在数据库中查到的id 和password是否对应，不对应就登陆失败。若对应，还需要先判断对应用户数据库中是否有hobby勾选，若无勾选则为新用户，进行兴趣爱好的勾选。有勾选跳到info主页面，开始推送。

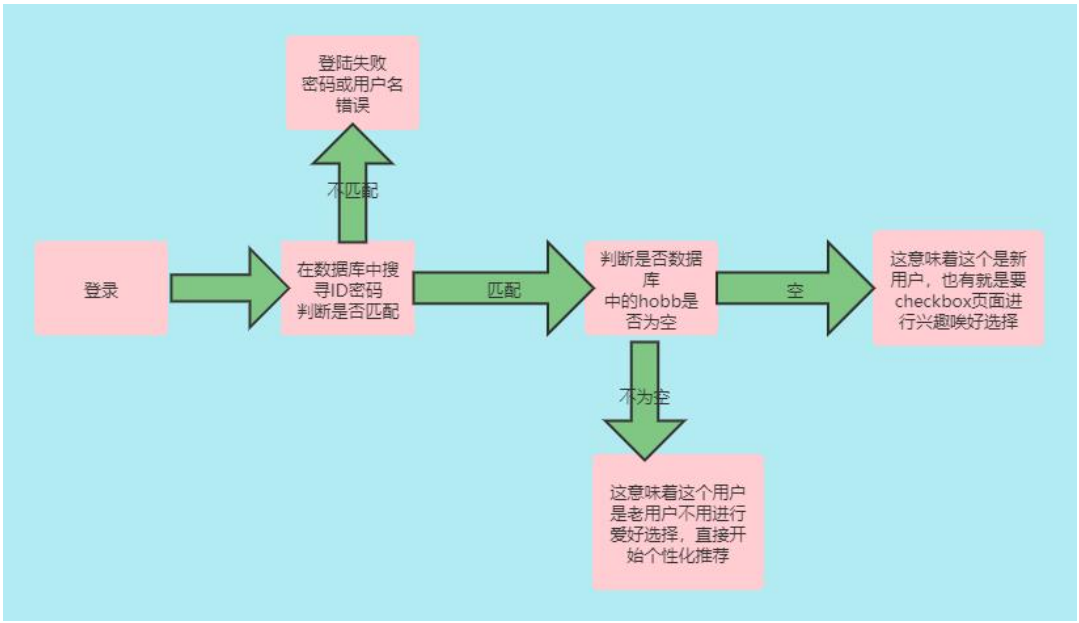


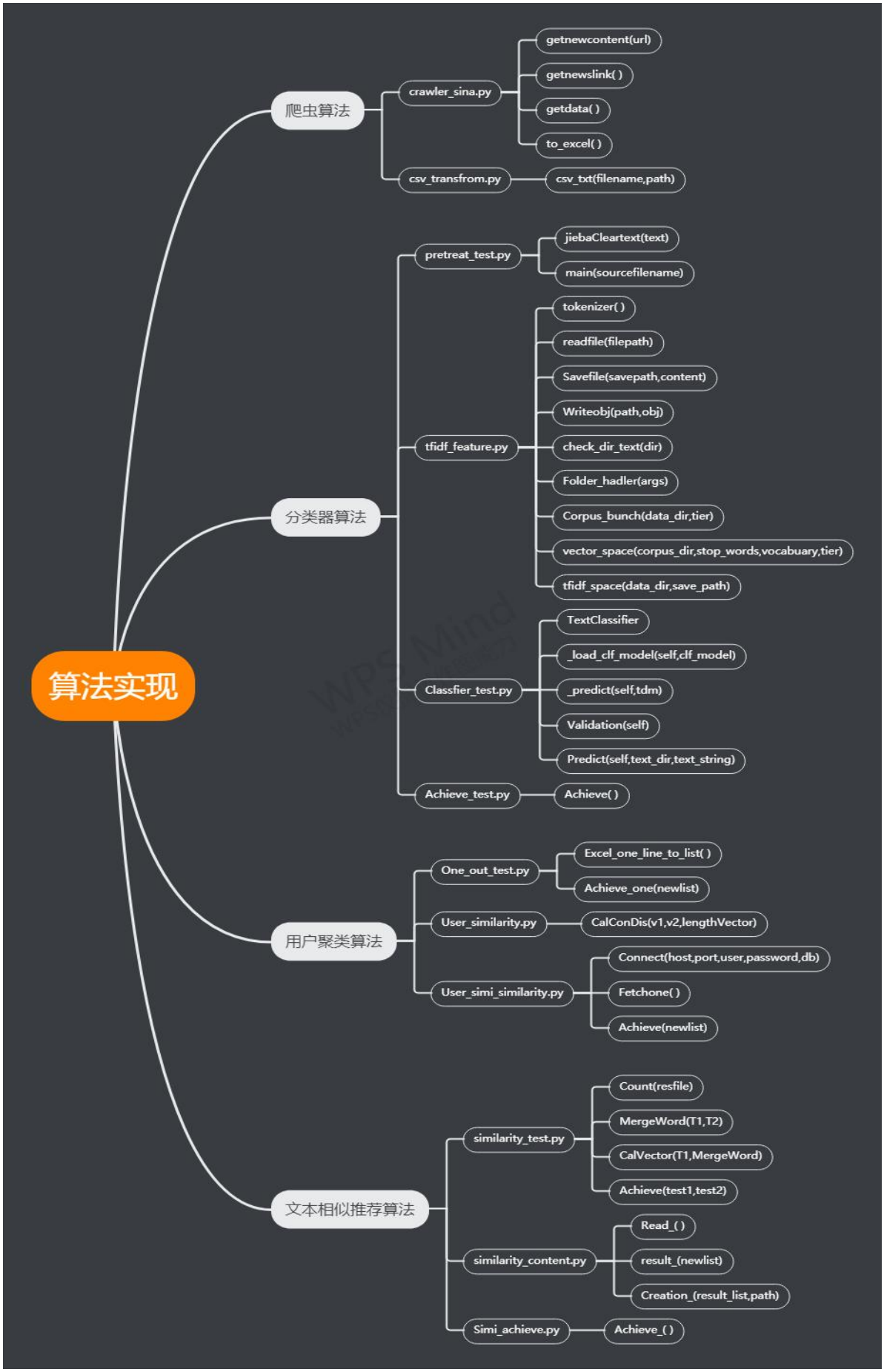
图3-6，运行框架图

CheckboxServelet：通过14个if语句，判断多选栏中的hobby是否为空（是否被选），若不为空则说明被选上。最后通过getRequestDispatcher跳转页面，跳转到主页面。

InfoServlet：这个部分是我们的主页面的实现部分了，这个算法（函数）调用了其他部分，把其他py文件都进行调用，实现了分析。首先是对页面进行打印，这个算法首先会打印一个页面，也就是个性化推荐的主页面。然后调用py文件进行个性化分析，聚类算法等进行对用户的服务。

3.4.2 算法部分函数分析

后端算法只要包括爬虫部分、分类器部分、用户聚类算法、相似文本推荐算法。其中爬虫部分是每天爬取先发表的新闻，并由此建立动态数据总库；分类器部分基于机器学习算法，对爬取的文章进行分类标签并添加到excel文件中；用户聚类算法是用于新用户首次登陆时挑选的兴趣标签，由此给出用户的推荐文本；相似文本推荐算法是根据用户的浏览日志，对用户的兴趣进行动态建模，并依次查询数据库中与其兴趣文本相似度较高的文本。


算法实现
爬虫算法
crawler_sina.py
getnewcontent(url)
getnewslink( )
getdata( )
to_excel( )
csv_transfrom.py
csv_txt(filename,path)
分类器算法
pretreat_test.py
jiebaCleartext(text)
main(sourcefilename)
tfidf_feature.py
tokenizer( )
readfile(filepath)
Savefile(savepath,content)
Writeobj(path,obj)
check_dir_text(dir)
Folder_hadler(args)
Corpus_bunch(data_dir,tier)
vector_space(corpus_dir,stop_words,vocabuary,tier)
tfidf_space(data_dir,save_path)
Classfier_test.py
TextClassifier
_load_clf_model(self,clf_model)
_predict(self,tdm)
Validation(self)
Predict(self,text_dir,text_string)
Achieve_test.py
Achieve( )
用户聚类算法
One_out_test.py
Excel_one_line_to_list( )
Achieve_one(newlist)
User_similarity.py
CalConDis(v1,v2,lengthVector)
User_simi_similarity.py
Connect(host,port,user,password,db)
Fetchone( )
Achieve(newlist)
文本相似推荐算法
similarity_test.py
Count(resfile)
MergeWord(T1,T2)
CalVector(T1,MergeWord)
Achieve(test1,test2)
similarity_content.py
Read_( )
result_(newlist)
Creation_(result_list,path)
Simi_achieve.py
Achieve_( )

函数名	所在文件名	函数功能	参数说明
getnewcontent(url)	crawler_sina.py	对于传入的url进行request解析,并获取相关链接的标题、日期、来源、链接、正文和作者	传入的参数为一个新闻平台的url。使用request获得url的接连解析，并且调用了BeautufulSoup进行获取html页面内容
getnewslink( )	crawler_sina.py	对于一个非标准的url链接，要首先解析url，以防止url本身加入了反爬算法，把输入的url转化为标准的链接	该函数无传参，解析url时直接调用执行，返回标准链接
getdata( )	crawler_sina.py	把全部爬取的文章所获得链接转化为标准格式并存储在一个列表中	函数返回一个列表，列表是爬取的各个文章的标准链接格式
to_excel( )	crawler_sina.py	将一个列表转化为xlsx文件，	使用时直接调用
csv_txt(filename, path)	csv_transfrom.py	将excel文件转化为文本，方便进行后续的基	Filename为excel文件所在位置，path为文

		于机器学习的分类实验	本输入的位置
jiebaCleartext(text)	pretreat_test.py	调用jieba分词库,自动对文本进行分词	传入的参数为待进行分词的文本
mian（sourcefilename)	pretreat_test.py	将传入的源文件进行分词之后，写入原来的文件	传入的参数为生成的文本，并且分词之后保存到的位置
tokenizer( )	tfidf_feature.py	返回jieba分词	无参数，使用时直接调用
readfile(filepath)	tfidf_feature.py	读取传入的文本	
Savefile(savepath,content)	tfidf_feature.py	把文本内容写入相关的地址	Savepath为保存文本的地址，content为写入文本的内容
Writeobj(path, obj)	tfidf_feature.py	持久化python对象,将要写入的信息加入到一个pickle中	Path为作为pickle第一部分的文本的内容，obj为pickle第二部分的内容
check_dir_text(dir)	tfidf_feature.py	查看是否存在该文件，不存在则创建	Dir为要观察的文件路径，不存在则创建
Folder_handler(args)	tfidf_feature.py	遍历一个文件夹下的文本，进行分词后，重新写入原文档	Args为要进行分词的文本的路径
Corpus_bunch(d	tfidf_feature.py	得到文本库，并返回一	data_dir为要进行获

ata_dir,tier)		个Bunch对象，里面的内容为根据测试文本所在文件夹附上的标签	取的文本的路径，tier为1或者2，当tier为1时，代表data-dir就是一级目录，里面是文本内容，当tier为2时，代表data_dir为二级目录，在该文件里面是各个文件夹，
vector_space(corpus_dir,stop_words,vocabulary,tier)	tfidf_feature.py	将一个语料库向量化，进行提取tf-idf特征，并返回一个带有特征词的向量空间	corpus_dir为传入的语料库，stop_words为去除的停用词，vocabulary为生成的特征词向量空间，tier判断路径是几级目录。
tiidf_space(data_dir,save_path)	tfidf_feature.py	将训练样本和测试样本提取特征之后合成一个vocabulary词库，并生成训练样本和测试文本的词库	data_dir为要进行提取特征生成词库的路径，save_path为生成的词库保存的路径，函数使用后会在目标路径下生成train_tfidf.data、

			test_tfidf.data、vocabulary.data
TextClassifier	Classifier_test.py	该变量为一个分类器的类名，里面包含分类器的各个函数	类名，无参数
_load_clf_model(self,clf_model)	Classifier_test.py	读取训练文本对分类器进行训练	clf_model为训练文本的路径
_predict(self,tdm)	Classifier_test.py	读取输入的测试样本的特征矩阵，返回根据分类器实现的预测类型	Tdm为测试文本的特征矩阵，返回一个该文本的预测标签
Validation(self)	Classifier_test.py	使用测试集进行模型验证，根据测试集原本所属的类型和预测的类型计算算法的准确率，召回率和正确率	直接对测试文本进行测试。
Predict(self,text_dir,text_string)	Classifier_test.py	对模型进行预测，观察分类器对测试文本所预测的结果是否正确。	text_dir为测试文本的路径，text_string为分类器对测试文本的预测结果
Achieve( )	Achieve_test.py	调用tfidf_feature.py和classifier_test.py，使用线性回归方程，对爬取的	直接调用，修改预测的excel文件即可实现

		文章进行分类，并把预测的标签前加到excel文件相对应的行中	
Excel_one_line_to_list( )	One_out_test.py	读取excel文件，并对指定转化为二维列表的形式	输出一个二位列表，记录所读取的内容
Achieve_one(newlist)	One_out_test.py	读取某个用户的数据库的选择日志，并对照总的类型把0、1序列转化为特定的标签，并使用倒叙删除为0的元素，只存储用户所选择的类型，然后对照excel总库中的列表，把类型相同的文件输出	Newlist为用户所选择的类型的0、1序列，输出为用户所选择的同类型的文件。
CalConDis(v1,v2,lengthVector)	User_similarity.py	计算v1，v2的余弦相似度，返回相似度的百分比	v1，v2为计算相似度的两个0、1向量，lengthVector为向量的长度
Connect（host,port,user,password,db）	user_simi_similarity.py	连接本地的mysql数据库，并执行sql语句	host为数据库的执行用户，port为端口号，db为连接的数据库

			的模式
Fetchone（）	user_simi_similarity.py	连接数据库，执行sql语句，返回执行结果，使用一个二维列表进行记录返回值	返回执行sql语句的的值
Achieve（newlist）	user_simi_similarity.py	首先判断数据库中有几位用户，若只有一位用户，直接按照他所选择的类型进行推荐，若有多位用户，读取新用户的id,并依次与数据库中的其他用户进行相似度计算，若相似度在[0.7,1]区间内内，认为相似度较高，使用一个新的列表记录二者都选择的类型，返回共同的类型文件，若无相似度较高的同类，则直接按照他所选择的标签反馈特定的标签	Newlist为连接数据库后记录的数据库中的信息
Count（resfile）	similarity_test.py	对分词好的文本去除停	Resfile为分词过后的

		用词，并统计关键词和关键词的词频，以列表的形式返回	文本，返回一个关键词和词频的列表
MergeWord（T1，T2）	similarity_test.py	合并T1，T2两个文档，得出一个共同文档，去除重复的词。	T1，T2为要进行合并的两个文档，返回一个合并的文档
CalVector（T1，MergeWord）	similarity_test.py	根据关键词向量和共同的向量，记录每个关键词的词频。得出文档的向量	T1为要进行统计词频的向量，MergeWord为合并的向量
Achieve（test1，test2）	similarity_test.py	对test1，test2文本进行预处理，统计词频，计算两个文本之间的相似度	Test1，test2为要计算相似度的两个文本
Read_( )	similarity_content.py	读取前端返回的用户id，在数据库中读取该用户的浏览日志，并存储该用户浏览网页的记录	读取前端传入的参数，返回相关用户的浏览日志
result_(newlist)	similarity_content.py	根据用户的浏览日志，在后端存储的数据总库中找到用户浏览的相关网页的信息	Newlist为用户浏览日志的列表，返回浏览网页的具体信息
Creation_(result	similarity_content.	根据用户浏览网页的具	result_list为用户所浏

_list,path)	py	体信息将相关的文本和数据总库的文本全部提取正文内容并将其转化为txt文档，存储在指定文档路径下	览的网页的具体信息，path为将正文转化为txt文档的输出路径
Achieve_( )	Simi_achieve.py	调用similarity_test.py和similarity_content.py两个文件中的相关函数，生成相关文本之后，再计算文本之间的相似度，把相似度在[0.5,1]之间的文本以指定的形式输出	直接调用实现，输出的为与用户所浏览的网页内容相似度较高的网页的相关信息

四、系统设计难点与亮点

4.1 前后端及交互部分

4.1.1 java 调用 python（.py 文件）实现大型程序框架。

因为小组成员熟练掌握的语言不同，我们主要使用 python 语言实现各种算法，然后用 java 语言连接数据库和主体框架，实现前后端的连接。总共有 3 种办法可以实现 Java 调用 Python 程序：

一.在 java 类种直接执行 python 语句

（1）此方法需要引用 org.python 包，需要下载 Jpython。（Jpython 引入百度百科解释：Jython 是一种完整的语言，而不是一个 Java 翻译器或仅仅是一个 Python 编译器，它是一个 Python 语言在 Java 中的完全实现。Jython 也有很多从 CPython 中继承的模 块库。最有趣的事情是 Jython 不像 CPython 或其他任何高级语言，它提供了对其实 现语言的一切存取。所以 Jython 不仅给你提供了 Python 的库，同时也提供了所有的 Java 类。这使其有一个巨大的资源库。）

（2）而且每次运行成功结果都会提示 console：Failed to install（类似于报错）

二.在 java 种调用本地 python 脚本

三.使用 Runtime.getRuntime()执行脚本文件

但在这里值得注意的是，前两个办法都可以调用 python 程序，但是都是使用 Jpython，调用的 python 库不是很多，同时我们的 python 文件调用了很多第三方库，所以我们 选择了第 3 个方法。

```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Demo1 {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Process proc;
        try {
            proc = Runtime.getRuntime().exec("python D:\\demo1.py");// 执行py文件
            //用输入输出流来截取结果
            BufferedReader in = new BufferedReader(new InputStreamReader(proc.getInputStream(
            String line = null;
            while ((line = in.readLine()) != null) {
                System.out.println(line);
            }
            in.close();
            proc.waitFor();
        } catch (IOException e) {
            e.printStackTrace();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```

图 4-1 java 调用 python 示例图

如图所示的代码解决了 Java 调用 python 程序的问题。

基本上可以实现 py 文件都用 Java 调用，因此我们可以把各个算法用 Python 完成，写成 py 文件，再有 java 来调用。

需要注意的是，不能在 python 中通过 return 语句返回结果，只能将返回值写入到标准输出流中，然后通过标准输入流读取 python 的输出值。

4.1.2 网页用户注册登录功能前后端连接

我们使用的是 html+mysql 实现的基本前后端的连接。.

第一步注册:

首先用户在 html 网页上输入信息并且此网页可以识别用户的账户名以及两次输入的密码是否符合规范。其次 html 网页通过某种途径把用户输入的账号、密码交给中继，中继再交后端数据库。为什么不能直接交给数据库呢？数据库不能实现处理

数据的功能，数据库只能保存数据，其他的任由别人摆布。

登录我们还从用户出发。不难想到就是将用户输入 html 的信息转送给数据库，借机完成查询，并可以返回是否查到的信息。与注册有共用的前后端连接的桥梁。

在这里注册登录界面我们要采用 form 标签来提交信息，action 是表单交给的 url，即 action 是用来指明这个中继的地址，method 是用来说明方法的。

在这里我们遇到了一个小问题，要注意到 input 的<type=”button”>和 button 的<type=”submit”>是完全不一样的前者没有提交功能，只是一个外表按钮，后者相反。

我们采取的是 java 来实现对 Mysql 数据库的增删查改，所以需要用 java 连接数据库。需要下载 JDK，mysql-connector-java 要和 mysql 版本匹配。

关键：中继 servlet 的实现

关键把以上两个部分都实现好。但没写好这个中继，我们的项目还是失败。

对 java servelet 的官方解释如下：

Java Servlet 是运行在 Web 服务器或应用服务器上的程序，它是作为来自 Web 浏览器或其他 HTTP 客户端的请求和 HTTP 服务器上的数据库或应用程序之间的中间层。

其次在注册的层面上来讲，中继是要拿到 html 的输入信息，并传给数据库。

需要对 doGet doPost 进行改写

最后使用的是 tomcat 服务器 9.0 版本的。版本太高不见得最好，与之匹配的没有做起来，会导致有不兼容的情况发生，所以我们尝试后选了个较高的版本，没有选择最新版本。需要把上文中提到 mysql-connector-java 的 jar 复制到 lib 文件夹种。否则会出现 http500 的错误。在最初创立 web 项目的时候一定要建立一个 web.xml 文件，webxml 文件要建立在 WEB-INF 下举个例子。

```
<!-- 配置 TestRegister-->
  <servlet>
  <servlet-name>testServlet</servlet-name>
  <servlet-class>cn.test.servlet.testServlet（包名.xml文件）</servlet-class>
  </servlet>
  <servlet-mapping>
  <servlet-name>testServlet</servlet-name>
  <url-pattern>/test.do</url-pattern>
  </servlet-mapping>
```

图 4-2 代码示例图

关于 servelet 的配置，servelet 中 servelet-name 是 servlet 的名字与<servelet>中元素的名字一致。

<servlet-mapping>将 URL 模式映射到某个 Servlet，即该 Servelet 处理的 URL。

4.1.3 html 页面跳转问题

我们将推送的网址打印在 html 的网页上，但是想实现上，一开始我们的页面跳转是在本页面跳转，但是我们想跳转到新的页面，这样在 html 那里需要修改， target=“_blank“这样的确可以实现新页面的跳转，但是跳转后却是空白页面，初步猜测可能是前面使用的 onclick 等 Javascript 的影响，所以我们退而求其次，用户在实现的过程中需要手动点击按钮。

4.1.4 用户画像存在问题

最初设计的时候我们是准备每次登录进入都要进行兴趣爱好的勾选，但是这样的话不是很符合逻辑，毕竟老用户已经有了浏览记录，也就是说根据算法可以对他的爱好有一个基本的用户画像，因此没必要每次进入都进行勾选，我们在登录处进行了优化。为此，我们单独设计了一个 ifΓableNull 函数，通过 While（res.next）循环，再进行判断 14

个标栏里面的是否为空，若为空我们就把它规划到新用户，新用户会强制弹出页面，进行兴趣爱好选择，这样才能首次刻画新用户的画像，进一步实现算法。由于算法相似度的计算是用余弦计算，如果没有用户画像，那么这样的用户向量为 0，不管和谁的乘积都为 0，同样也会造成算法失效，因此我们在登录页面就避免新用户没有用户画像进入数据库。

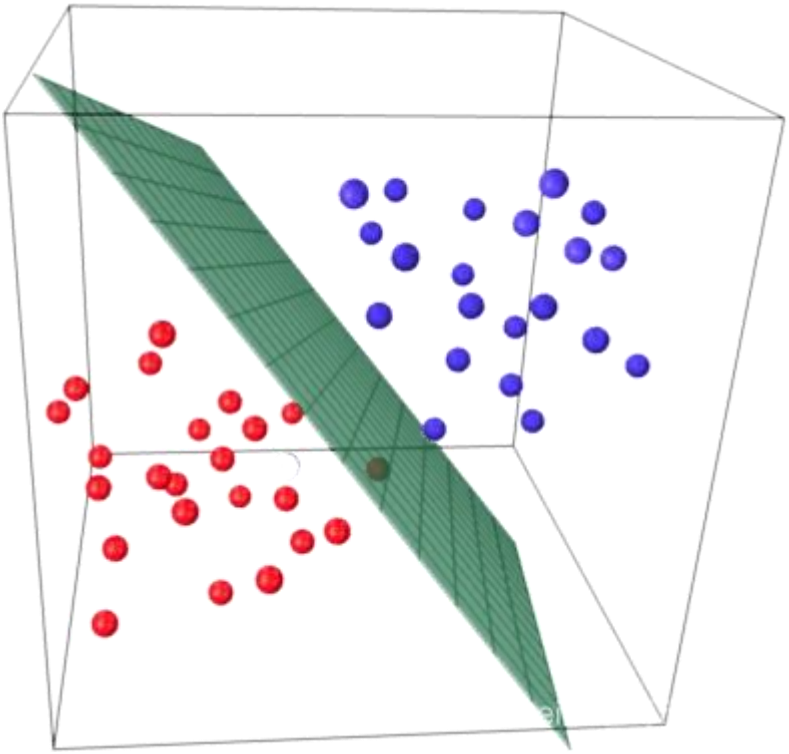
4.1.5 算法原理

1. 分类器

分类器的作用：常规任务是利用给定的类别已知的训练数据来学习分类规则和分类器，然后对未知数据进行分类（或预测）。

我们采用的是线性回归分类器

线性分类器背后的基本思路是，目标分类的值可以被特征空间中的一个超平面分开。如果这可以无误差地达成，那么训练集被称为线性可分。



最简单的线性分类器可以通过回归定义：

$$a(\mathbf{x}) = \text{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x})$$

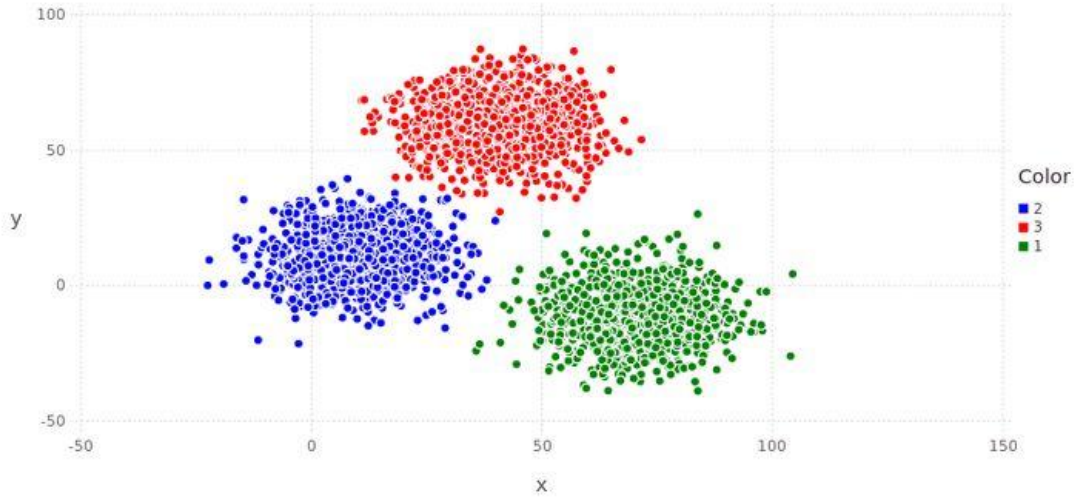
其中：

- $\mathbf{x}$ 是特征向量（包括标识）。
- $\mathbf{w}$ 是线性模型中的权重向量（偏置为 w_0）。
- $\text{sign}(\bullet)$ 是符号函数，返回参数的符号。
- $a(\mathbf{x})$ 是分类 $\mathbf{x}$ 的分类器。

基于上述的定义和条件，可以说，根据高斯-马尔可夫定理，模型参数的 OLS 估计是所有线性无偏估计中最优的，即通过 OLS 估计可以获得最低的方差。

2. 用户聚类

基于上述的定义和条件，可以说，根据高斯-马尔可夫定理，模型参数的 OLS 估计是所有线性无偏估计中最优的，即通过 OLS 估计可以获得最低的方差。



3. K 均值聚类

K 均值（K-means）聚类是一种迭代的聚类算法。K 均值聚类要求在建模初期确定聚类簇（Cluster）的个数。由于我们知道本问题涉及 3 种花的类别，所以我们通过将参数 n_clusters=3 传递给 K 均值模型来实现聚类。在实际场景中，也可以根据业务需求，进行多次尝试，最后确定最符合业务目标的聚类数量。需要注意的是，K 均值聚类需要对用户的所有维度变量的量纲进行统一，比如进行归一化或标准化

原理：在 K 均值聚类的过程中，首先随机地将 3 个数据点分到三个簇中，并将该 3 个点视为当前簇的质心。基于接下来每个点到这 3 个初始点之间的质心距离，确定下一个给定的输入数据点将被划分到哪一个簇中。待所有点归类结束，重新计算所有簇的质心，然后再次计算每一个点到质心的距离。该过程不断重复，直至满足收敛状态。

均值漂移聚类计算过程：

1）设想在一个有 N 个样本点的特征空间

2）初始确定一个中心点 center，计算在设置的半径为 D 的圆形空间内所有的点（xi）与中心点 center 的向量

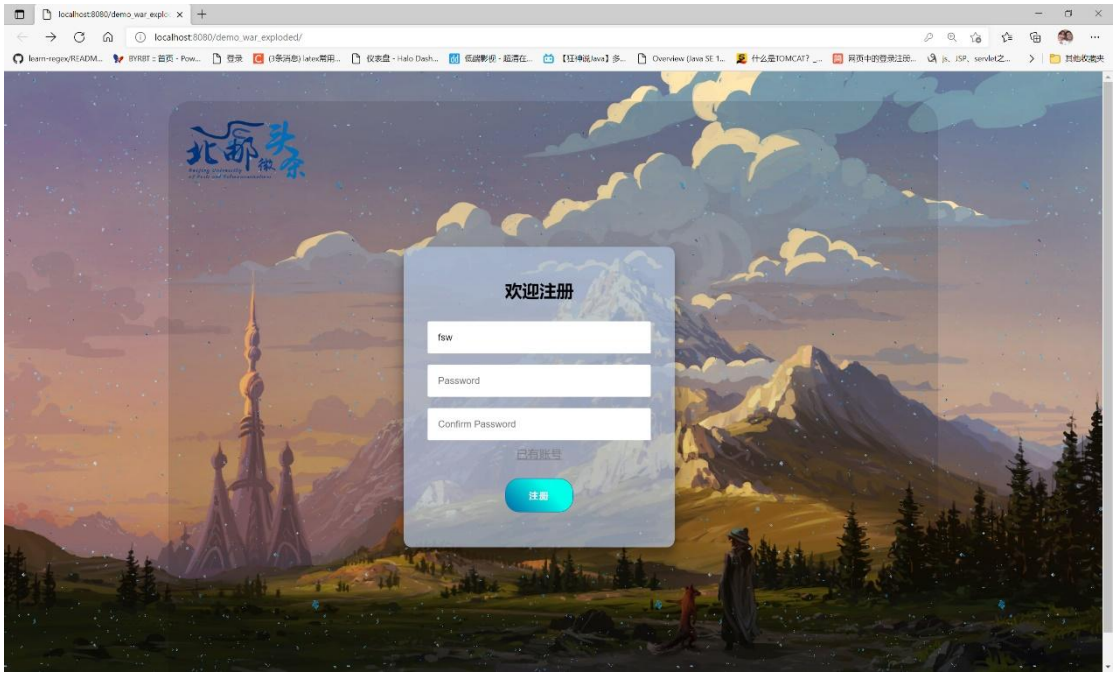
3）计算整个圆形空间内所有向量的平均值，得到一个偏移均值

4）将中心点 center 移动到偏移均值位置

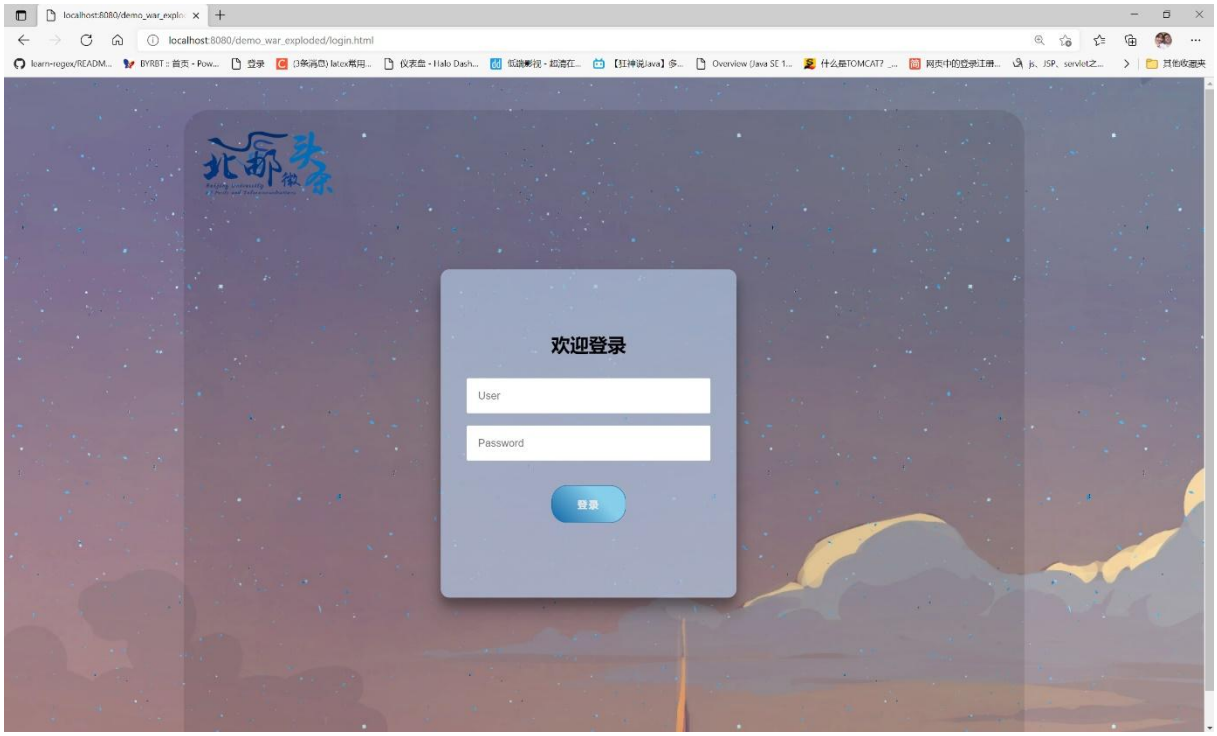
5）重复移动，直到满足一定条件结束

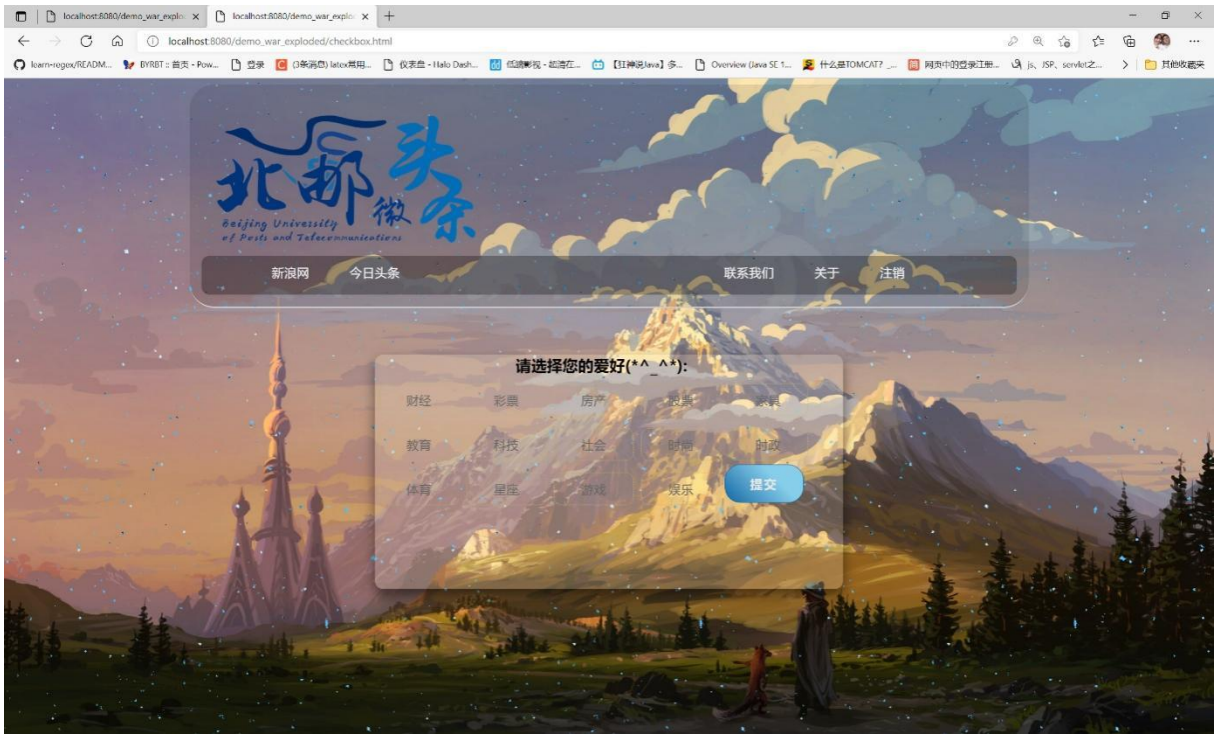
K 均值聚类对噪声的鲁棒性没有均值漂移聚类强，初始点选择不当会导致 K 均值聚类陷入局部最小值问题，且均值漂移聚类是一个单参数算法，容易作为一个模块和别的算法集成。因此，我们可以使用均值漂移聚类对数据集进行初步聚类，从而达到去噪的目的，然后再使用每一簇的质心代表整个簇，通过 K 均值聚类得到指定数量的簇

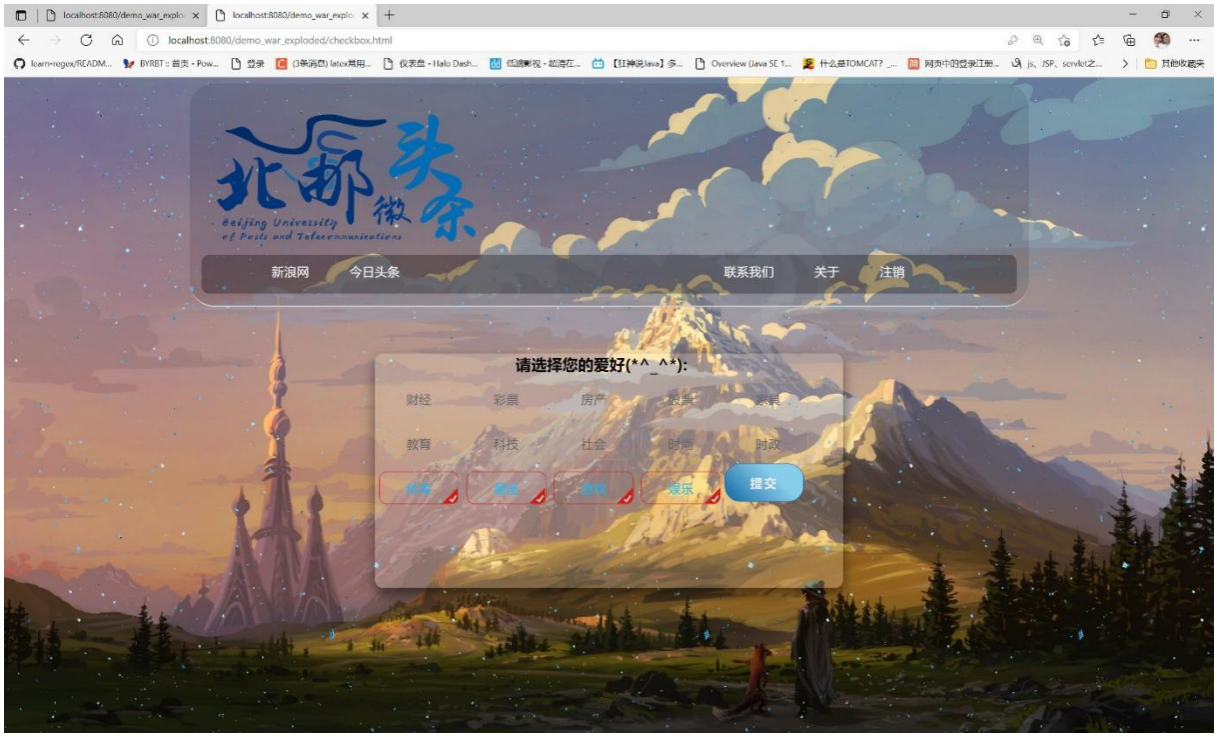
五、设计成果


localhost:8080/demo_war_exploded/
欢迎注册
fsw
Password
Confirm Password
已有账号
注册

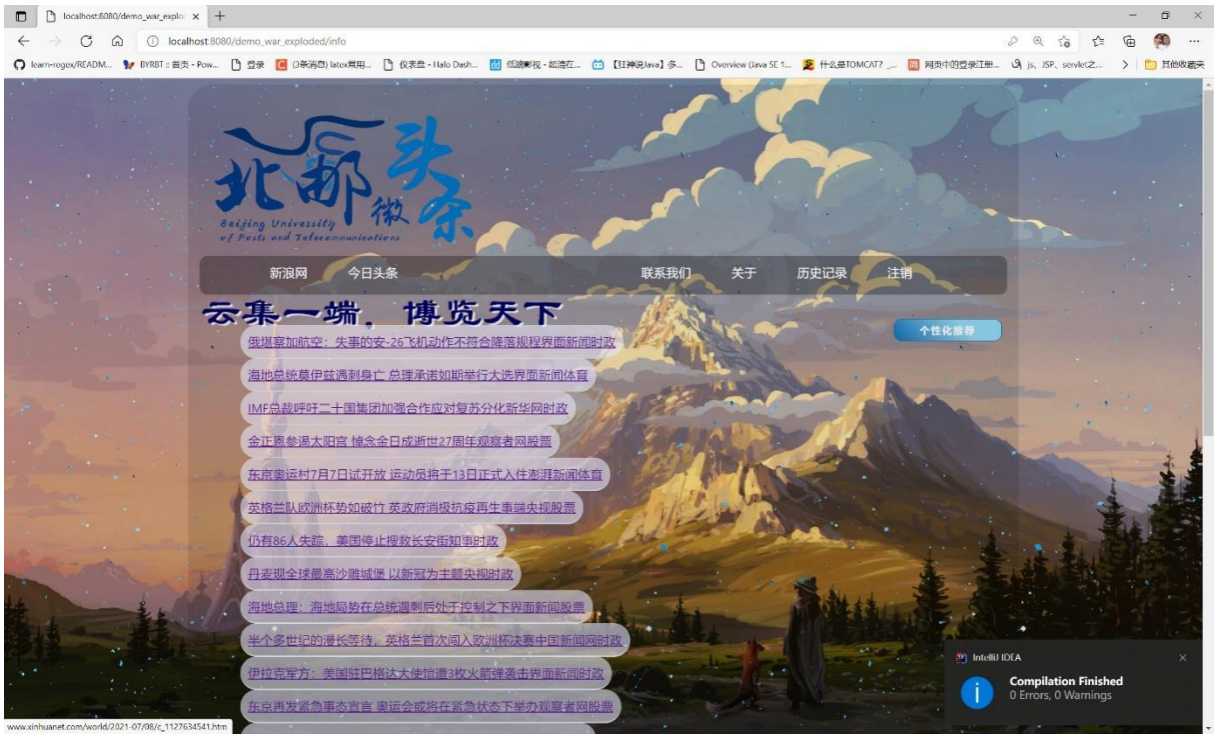
以上是注册以及注册成功的成果展示



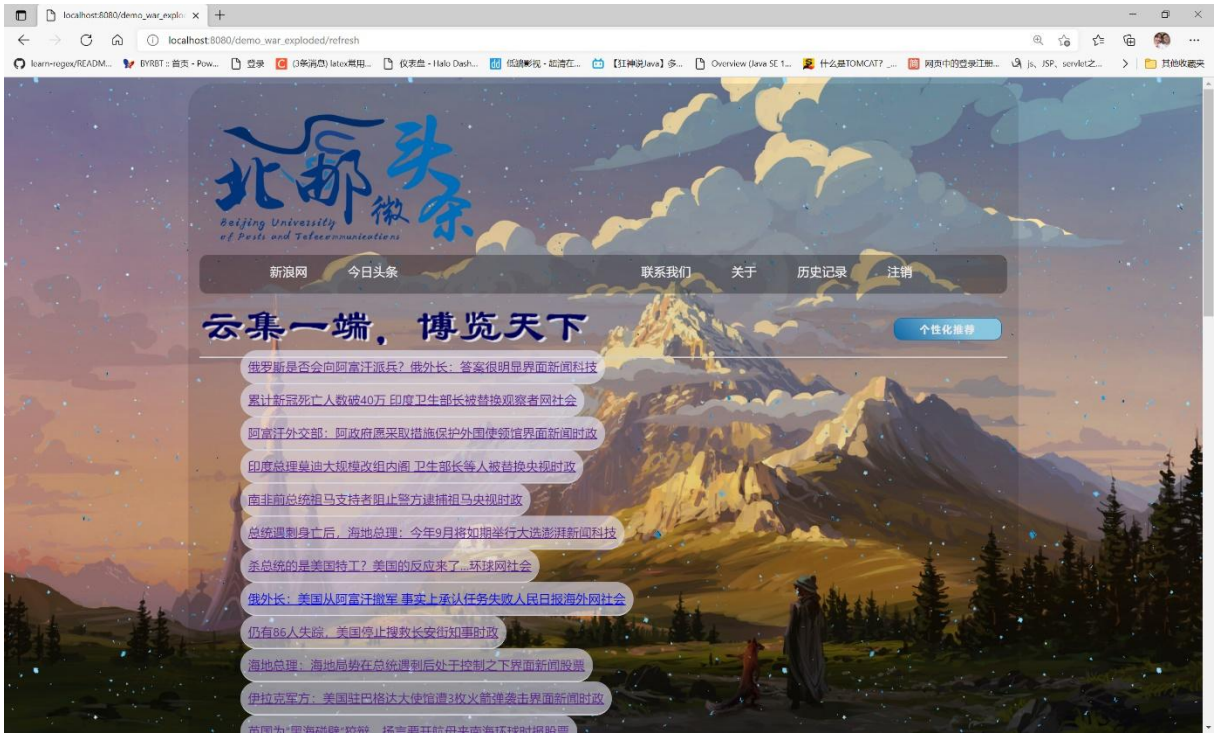




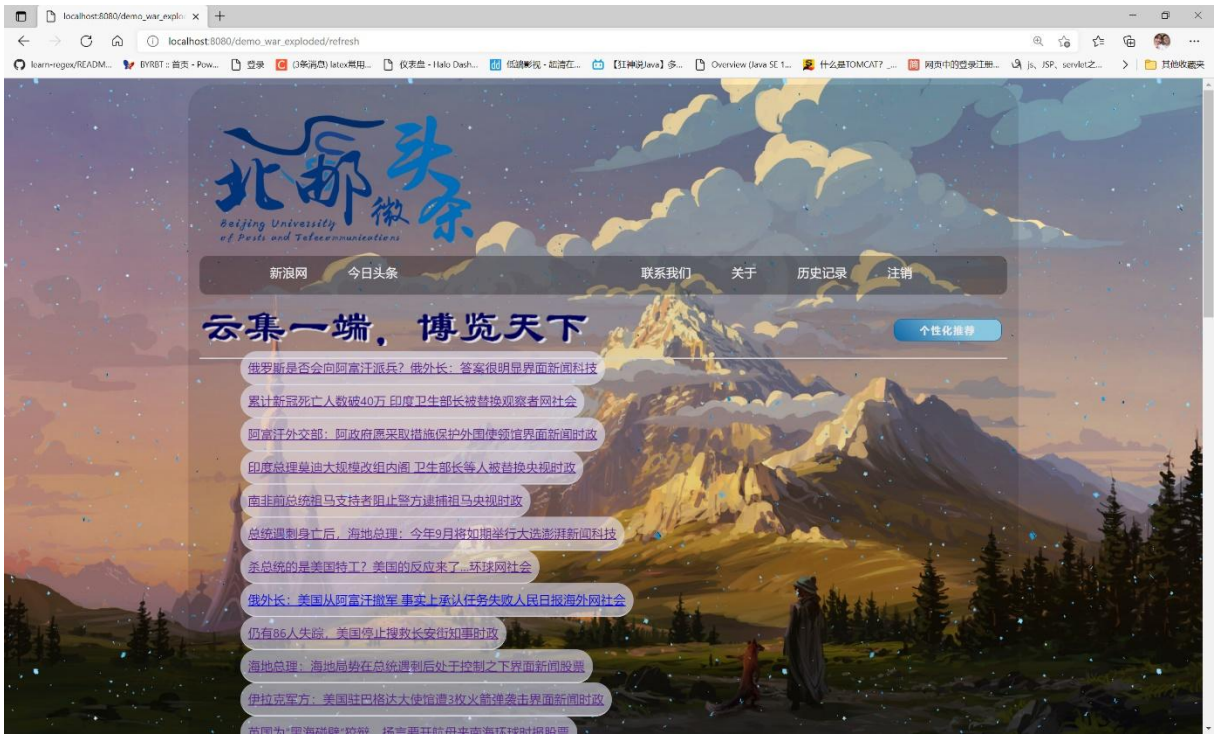
以上是新用户登录，要进行初始化所以爱好选择。



第一次登录的时候根据兴趣爱好的推送。



第一次登录的时候根据兴趣爱好的推送。



第二次，有了浏览记录，就实现了兴趣漂移，再次点击个性化推荐，发现推送的内容又发生了变化

六、遇到的问题及解决

6.1 前端部分

网站制作的 web 前端开发技术发展以来，web 前端开发技术最　开始使用 HTML 技术实现网页内容的访问、css 技术实现网页站点的构建和页面的美化。作为将功能、设计理念呈现给用户的重要窗口，前端开发承担　了较大的压力。在与后端人员详细确认功能模块、呈现方式、整体风格之外，还要站在用户的角度，组织页面之间的逻辑性、合理性、可用性。在完成基本页面设计的基础上，web 前端开发的研究中以客户反应和客户体验为重点研究方向和完善内容。

6.1.1 前后端分离问题

在着手项目的开始,我们全小组的成员由于都没有做过网页开发和程序设计的经验,导致初期的工程分工不够清晰。在模糊的印象中将人员大致分成了前端开发人员 2 人,后端开发人员 1 人,算法开发人员 2 人。算法设计时使用 python 语言实现,前端使用 html+JavaScript+CSS,而后端人员运用 JavaWeb 技术实现。由于 JavaWeb 技术有些古老且技术基础浅薄,导致前后端未在真正意义上未分离。

后在项目第二天,由于各部分工作量问题,调整人员为前端开发人员 2 人,后端开发人员 2 人,算法开发人员 1 人。前端人员计划使用 Vue 框架+Element Plus 来实现前端部分,希望使代码和函数更加规范。而在与后端人员讨论交流后,发现使用 JavaScript 语言将大大增加前后端对接问题。

经过多次协商,为项目整体考虑,前端人员放弃了前面所有的工作成果,转而使用 html+CSS 实现前端部分工作。并且在最后阶段的前后对接时,前后端人员对每一个对接后影响效果的问题进行了解决。

6.1.2 经验问题

因缺乏 web 开发的经验,前端成员经学习和准备,配置环境,采用了“html+css+js”的语言,基于 vue3.0 框架,辅助以 element plus 的前端 UI 库,进行比较规范的前端开发。前端思路的理想化方案为,采用前后端分离的思想,前端可以有更大的空间来进行创新性的设计、实现一些炫酷的效果,并进行一些逻辑上的组织来减少后端的工作量。但在开发进行一段时间后,发现因交互数据受阻。经紧急讨论,小组成员决定以实现功能为第一目标,将交互主动权掌握在后端,前端被迫放弃原本的思路和技术路线,改为不断适配后端,提供多种设计方案,并对可以适配后端的 demo 进行扩展、美化。小到一个

组件的位置，大到整体布局的有序、一个设计效果的展示，都需要前端同学劳神费力地调整。

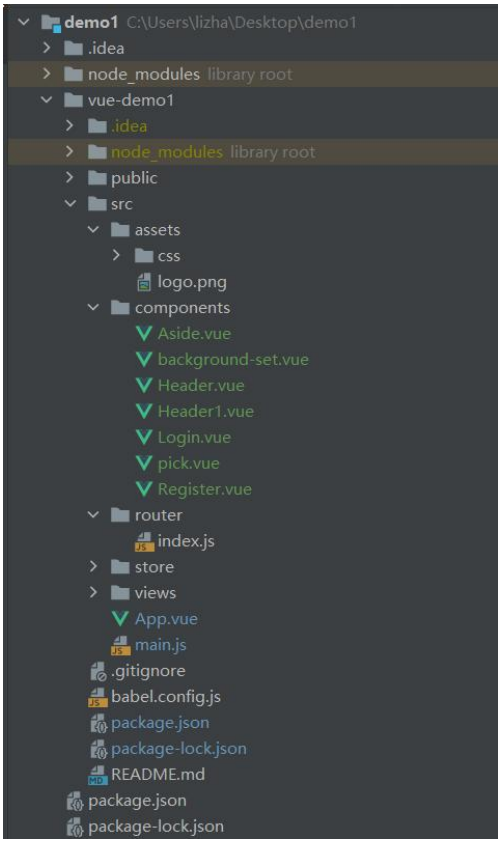
在这个过程中，前端同学化整为零，优化显示效果，力求将显示效果做的更适合对应功能模块；不断调整显示上的 bug，解决布局、样式上在不同设备上出现的问题；要在其他成员的建议下、在后端的实现方式下，动态变化需求，改进我们的呈现方式。在不断的修改过程中，我们更加深刻地领悟到了“设计”和“规划”的重要性，不提前确定所有细节，就会在不断地修改和重做中无谓消耗时间和精力。

6.1.3 其他细节问题

由于之前对 html 和 CSS 语言没有基础，在页面设计的过程中碰到了许多语法及排版细节问题。我们通过上网查阅资料和向老师同学交流询问解决了问题。

例如在配置 Vue.JS 框架的环境时，由于需要预先安装 node 和 npm，并启动创建 Vue 文件，出现了许多问题，我们花费了大约一个早上的时间去配置编写代码的环境。

从单纯的 html+JS+CSS 到计划使用 VUE 框架增强项目的规范性，再到为整体效果和结果成功率考虑，转到只能使用 html+CSS 的静态网页。对于零基础的我们来说，或许并不是一件容易的事。



（idea 环境中，Vue 框架的学习和尝试过程）

6.2 后端交互层

6.2.1 系统框架

后端最大的问题主要是在于框架，开始的时候不知道怎么用搭建起整个系统，所以我们查阅资料，参考了别人的体系。最终的实现是使用 html+mysql 实现基本的前后端连接，连接数据库的主体部分等都是用 java 实现的。这一部分都是我们一步步摸索，没有使用其他框架的，所以这一部分的搭建耗费了大量的精力和时间。在这当中，还有一些 servelet 中继的问题，我们使用的是 IDEA 编译器，和网上资料的 eclipse 不同，因此我们一步步百度搜索解决 servelet 的配置，最终成型。

6.2.2 语言问题

第二个摆在我们眼前的问题就是语言的问题，因为开始的时候我们只想先实现后端基本的注册登录功能，但是这个过程就花了大概一天半的时间，我们就继续选择原来的道路，继续使用 java 语言做下去。但是我们的算法设计使用的 python 语言，这样就有语言问题了。

最初的问题是算法是 python 输出的结果要怎么写到我们的 html 文件上，但是因为我们前后端的连接还有 java。因此，我们计划是在 python 算法上改进，输出是一个 txt 文件，再用 java 读取 txt 文件，用 java 再进行输出到 html 上。

但是这个方法很难查询到相应的资料，并且实现的过程比较繁琐复杂，因此在这里我们又卡住一段时间。

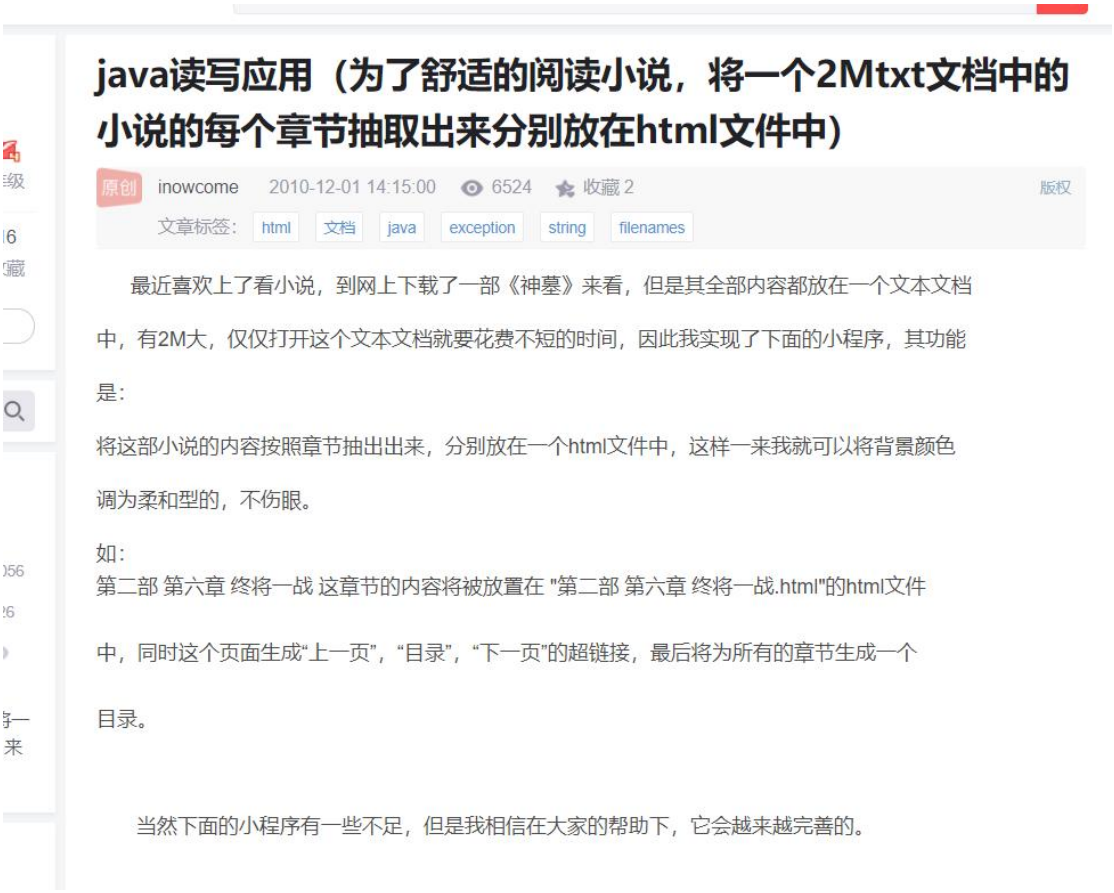


图 7-1　java

（查到的帖子都已经是 2010 年的了，十多年前的技术了，现在 java 都不试用了）

我们查询资料，找到了好几个办法，但是因为算法上实现使用了很多第三方库，这样就不得不只使用 Runtime.getRuntime（）执行 py 脚本文件。这个是 java 调用外部可执行程序或系统命令，并重定向外部程序的标准输入、标准输出和标准错误到缓冲池。也就是说这个函数的运用是外部调用 py 文件，也就是在 cmd 中也能运行。因此我们还需要配置 python 到全局变量，才能完美实现 Java 调用 python。至此，我们已经大体上理论可以实现了前后端的连接，和后端算法调度的问题。

6.2.3、乱码问题

这个问题的出现是在页面跳转的时候出现的。我们最初使用的跳转函数是 getrequestdispatcher。通过查询，我们发现，要在该函数前面设置编码，setContentType（“text/html;charset=utf-8）或者 response.setCharacterEncoding("UTF-8")，而且有的资，两者只能有一个，但是这样修改后乱码依然存在，把 UTF-8 改成 GB2312 依然会出现乱码，最终这个问题并没有找到一个好的解决方案，只能将，getrequestdispatcher 改成最后使用的 endRediretct 才能解决。

6.2.4　404，500 等未响应问题

这一类问题是后端出现的最多的问题，但是每一次的 404 500 都可能是由不同的原因出现的。比如说 500 最大的一个问题，也是困扰我们最久的一个问题是，要把 mysql-connector-java 中的 jar 复制到 Tomcat 的文件加中。（lib 是调用包的文件夹集合。）

6.2.5、逻辑上的系统 BUG

我们设计的系统是在每一次登录都需要对爱好进行选择，但是这并不符合逻辑，因为老用户根据浏览历史，已经由算法分析计算出了用户画像，而新用户才是需要解决用户画像的问题。同样的，我们发现，有的时候在调用算法的时候会出现除零异常，回到数据库中，才发现，若一个用户的所有兴趣 hobby 都为 0 的话，（也就是没有兴趣爱好）会出现这样的错误。这是因为 python 算法的向量为 0，没法进行计算因此我们后端准备从系统层面解决此问题。我们设计在登录界面的时候就会对该用户进行是否是新老用户进行一个判定。这个判定依据也很简单，并不是依据你是否是第一次登录来检测，而是以是否有兴趣爱好的来实现的。如果不是老用户就会进行个性化推荐，如果是新用户就有兴趣爱好勾选页面跳转。这个检测是在登录的时候进行的，最开始的登录仅仅只是对密码和用户名进行判断是否匹配，现在我们还会对后续的表单进行检验，如果有一个用户的爱好全都是 0 就会被判断为新用户。这样我们既避免了算法上的 BUG 又解决了逻辑上的 BUG。

6.2.6 兴趣漂移的解决

兴趣漂移算法在移植上一直都有些问题。兴趣漂移是在算法上本身就调用了多个 py 文件，而我们后端还需要用 java 调用 py 文件，这样又多了一些问题。首先是接口的问题。因为算法和后端没有沟通好，导致接口一直不匹配，程序在运行的过程中也一直在报错。但是关键问题是每次运行都不会出结果，将算法用 CMD 运行会发现报错超出 index 数组的范围。最后检查发现是数据库中的浏览历史不匹配，也就是说第一次的浏览记录被保存在了数据库中，但是我们后端中途更换了一次爬虫爬取的各种链接的库，导致原先部分的数据，无法找到，这就导致了数组越界的报错。

七、设计心得

忙碌了一个星期，在大家的共同的努力下，我们总算设计出了此程序。在此过程中，我们从开始的毫无头绪，四处查询资料，到初步的分工合作，一点点的开始，再到后期的完善功能，大家都付出了很多的心血。首先，我们对大型程序设计有了基本的概念。首先是前端的页面设计，我们五个人基本都没有什么基础，但是通过一个星期的尝试和学习，对前端的几个语言 html，css，JavaScript 都有了初步的认识，同时可以基本写出一些简单的页面。

在知识的运用上，我们在大二上学期学习了数据库相关知识，但是大多都是一些理论上的知识，没有相应的运用。这次因为登录注册页面，我们开始使用了 mysql，数据库相关知识不得不运用在这上面。同时我们还要用 java 连接数据库，通过 Java 对数据库操作。其实最主要的问题还是在前后端交流的问题上。我们经常出现前后端目标不明确，因此不得不走了很多弯路，核心问题和难题在后端，因此，我们在前端的开发上都是以后端为主，根据后端的要求来进行前端页面的设计。

前端部分：

前端做为与用户交互的第一媒介，承担了印象初体验的重任。此次程序设计中，前端开发人员在技术上,从单纯的 html+JavaScript+CSS 转到预期效果更好的 VUE 框架+JS，再为对接成功完成率和整体效果考虑，牺牲了前些天完成的 vue 文件，而转至由 html+CSS 实现静态网页。

虽然最终呈现的成果对于我们来说可能仅有这些天努力结果的四分之一或者五分之一，但是我们在过程中收获了以前从未接触过的新知识，也对未来公司中项目的形成，精美界面的生成有了清晰的概念。

后端部分：

由于后端开发的两人都是第一次操作，经验不足，并且大部分的代码都是自己上网自学，再自己手工实现的，而且没有使用框架，因此对于部分技术上不是很了解。比如JavaScript 部分，这一部分不得不限制了前端的操作，因此我们的前后端分离情况并不是很理想，二者都有所约束，我们对前端要求尽量少使用 JavaScript 的内容，因为涉及到JavaScript 的数据接收回来我们处理的不是很理想。其次是我们再开始的前两天走了很多弯路。我们的后端在解决问题的时候一直没有想到合适的办法，再查询资料的时候寻找到了一篇 java 为语言的，html 加 mysql 组成的登录注册为模板的资料，因此我们也模仿着去实现，这就导致了我们的大体框架是以 java 为核心语言，但是我们的算法部分却又是由 python 来实现的，最终导致了我们使用了 java，python 双语言，所以我们后端又要想办法解决两个语言共存的问题，最终我们还是查到了相应资料，使用了 java 语言调用 py 文件的办法，最终这个问题也算是解决了。但是通过本次的大型程序设计，我们对后端也有了一定的了解，也有了相应的经验，通过观察其他组，我们也学习到了先进的框架模板知识，虽然过程艰辛，但是却收获颇丰。

八、团队分工说明

汤博：算法端开发核心人员、主要文案撰写、PPT 制作

卢潇：后端开发核心人员、负责各部分对接

范世炜：后端开发人员、文案撰写、主要 PPT 制作

孙泰城：前端开发核心人员、文案撰写

李兆腾（小组组长）：前端开发核心人员、统筹规划、PPT 制作