

**Detailed project plan**

**项目详细方案**

人才市场大数据智能分析展示系统

2019

**目录**

**[第一章 前言 5](#_Toc904872176)**

[1.1项目概述 5](#_Toc1854957625)

[1.2项目背景 5](#_Toc1252699876)

[1.3编写目的 6](#_Toc197140744)

[1.4预期读者和阅读建议 6](#_Toc1924700734)

[1.5参考资料 6](#_Toc899061577)

**[第二章 需求分析 7](#_Toc832984347)**

[2.1 系统功能性需求分析 7](#_Toc522025236)

[2.2 系统非功能性需求分析 7](#_Toc1207443457)

[2.2.1 用户界面设计规则 7](#_Toc1929201296)

[2.2.2 外部接口 7](#_Toc1378079466)

[2.2.3 出错处理设计 7](#_Toc770452167)

[2.2.4 系统维护设计 8](#_Toc1810663006)

**[第三章 概要设计说明 9](#_Toc1969863852)**

[3.1设计目标与原则 9](#_Toc1893858412)

[3.1.1设计目标 9](#_Toc75714650)

[3.1.2 建设原则 9](#_Toc1225803526)

[3.2 运行环境 10](#_Toc1269235811)

[3.2.1硬件环境 10](#_Toc1091209826)

[3.2.2软件环境 10](#_Toc453200202)

[3.3 系统总体架构 11](#_Toc1958782752)

[3.4 系统功能设计 12](#_Toc337404354)

[3.4.1 动态展示模块 12](#_Toc1398149598)

[3.4.2 智能分析模块 14](#_Toc934228112)

[3.4.3 科学预测模块 15](#_Toc1318935167)

[3.4.4 个性推荐模块 16](#_Toc1017147435)

[3.4.5 数据后台管理模块 16](#_Toc1227109925)

[3.4.6 移动端控制模块 17](#_Toc1751047334)

**[第四章 技术路径综述 18](#_Toc736644050)**

[4.1系统总体流程 18](#_Toc533323395)

[4.2第一阶段：数据采集（基于聚集爬虫的数据采集） 19](#_Toc2117040834)

[4.2.1聚焦爬虫工作过程 19](#_Toc1596233542)

[4.2.2 登录验证 19](#_Toc1531422070)

[4.2.3 URL管理 19](#_Toc1019221195)

[4.2.4 页面获取 20](#_Toc1721055893)

[4.2.5 页面解析 20](#_Toc1329152208)

[4.2.6 数据清洗 21](#_Toc936263762)

[4.2.7 数据存储 22](#_Toc1172366365)

[4.2.8 补充说明 22](#_Toc799035330)

[4.3第二阶段：数据预处理（基于Hive的数据预处理） 23](#_Toc1171546619)

[4.3.1 基于Hive的数据预处理过程 23](#_Toc2053949837)

[4.3.2 数据抽取 23](#_Toc2082768581)

[4.3.3 数据探索分析 23](#_Toc1108094767)

[4.3.4 数据预处理 23](#_Toc770562185)

[4.4第三阶段：数据分析①(基于Hive的离线数据分析) 25](#_Toc1512251885)

[4.4.1 Hive进行离线数据分析的优势 25](#_Toc948468950)

[4.4.2 Hive数据仓库基本架构 25](#_Toc146530969)

[4.4.3 离线数据分析设计概要 26](#_Toc1729736521)

[4.5第三阶段：数据分析②(基于HBase的实时数据分析) 27](#_Toc1195579008)

[4.5.1 Hbase技术介绍 27](#_Toc91902477)

[4.5.2 HBase系统架构 27](#_Toc564188746)

[4.5.3 HBase读取数据流程 28](#_Toc1179952517)

[4.5.4 数据Flush过程 29](#_Toc1597956821)

[4.6第三阶段：数据分析③(基于Spark MLlib算法的信息预测) 29](#_Toc429801165)

[4.6.1 Spark MLlib机器学习库介绍 29](#_Toc1680675294)

[4.6.2 线性回归算法原理分析 30](#_Toc1257257267)

[4.6.3 关于梯度下降算法的选择 31](#_Toc1631283636)

[4.6.4 线性回归算法LinearRegressionWithSGD 32](#_Toc60349003)

[4.7第三阶段：数据分析④(基于内容过滤和协同过滤的推荐算法) 36](#_Toc673412037)

[4.7.1 混合推荐算法技术特色 36](#_Toc797286169)

[4.7.2 基于内容过滤的推荐模型 37](#_Toc1838168750)

[4.7.3 文本分词 37](#_Toc402435508)

[4.7.4 文本相识度计算 38](#_Toc1307578553)

[4.7.5 匹配度计算 39](#_Toc1272580520)

[4.7.6 基于双重聚类的协同过滤推荐 40](#_Toc1471159167)

[4.7.7 双重聚类的协同过滤推荐 40](#_Toc1792891858)

[4.7.8 个性推荐总体设计 44](#_Toc1790406349)

[4.8第四阶段：数据导出 45](#_Toc818645879)

[4.8.1 数据导出流程 45](#_Toc53562024)

[4.9第五阶段：数据可视化 45](#_Toc421289275)

[4.9.1 通过Bootstrap进行数据可视化的优势 45](#_Toc355260766)

[4.9.2 ECharts图表展示 46](#_Toc863155502)

[4.9.3 ECharts图表展示的流程 46](#_Toc802486629)

[4.9.4 Ajax数据交互 47](#_Toc1195470443)

[4.9.5 Ajax进行数据交互的流程 47](#_Toc414734169)

[4.10第六阶段：系统管理①（数据后台管理模块） 48](#_Toc1852743868)

[4.10.1 bootstrap框架介绍 48](#_Toc553307976)

[4.10.2 SSM框架介绍 48](#_Toc842961122)

[4.10.3 SSM 框架流程介绍 48](#_Toc697958195)

[4.11第六阶段：系统管理②（基于Android的移动端控制模块） 49](#_Toc1027703451)

[4.11.1 Android介绍 49](#_Toc400928136)

[4.11.2 Butterknife模版代码生成工具 49](#_Toc1742981113)

[4.11.3 Rxjava响应式编程 50](#_Toc459137464)

[4.11.4 Glide图片加载库 51](#_Toc814613777)

[4.11.5 Android Library 51](#_Toc1005500414)

[4.11.6 网络编程技术 52](#_Toc896639855)

**[第五章 详细设计说明 54](#_Toc933291986)**

[5.1 用例图 54](#_Toc617851014)

[5.2 顺序图 56](#_Toc1138559053)

[5.3 数据输入信息表 59](#_Toc1682709001)

# 第一章 前言

## 1.1项目概述

* 软件的系统名称：人才市场大数据智能分析展示系统
* 项目面向用户：企业HR人力资源部门 、高等院校就业处、人才市场求职者
* 项目开发者：DataScreenPlus团队

## 1.2项目背景

* **整体背景：**

本题目要求设计并开发出一个网页版大数据可视化交互系统，其中包括一个在移动终端（手机或平板电脑）上使用的设置和控制界面，一个在大屏幕（LED大屏或电视）上使用的大数据展示系统。数据可视化是大数据生态链的最后一公里，也是用户最直接感知数据的环节。大数据技术正在飞速发展，从行业上看，智慧城市等智慧业务在当前市场发展很快；目前大数据产品已经比较成熟，并广泛应用于互联网管理部门、部队、企业和高校；另外还有商业智能、工业监控等领域。以前很多人只听说过大数据，但没有见过，随着越来越多的行业应用了大数据，通过大屏幕展示大数据，才知道原来这就是大数据。比如，智慧城市可视化平台，通过将政府智能部门（市政、交通、消防等）所有的数据汇聚到系统中，然后展示出来，就可以直观的看到整个城市的数据态势，还可以看到数据的具体变化情况。人是比较抗拒枯燥数字的生物，用可视化方式呈现出来，会更易于接受、理解并记住。 对于大多数大数据应用，前期针对数据的采集、传输、存储、分析、运算等工作，都是基础性工作，必须通过可视化的手段，将各类数据当中的规律和联系展现在决策人面前，才能让数据有效支撑最终的决策过程。

* **业务背景：**

用户需求源于在某单位指挥中心建成的一个大数据大屏交互系统。比如指挥中心拥有一面宽 16 米，高 3 米的集成（16：3）LED 屏幕，现场有一个集成了视频、音频、网络交换、中控服务器的机柜，可联通一个已建成的大数据服务中心。指挥中心的数据由大数据服务中心和接入现场的音频、视频提供。屏幕的控制需要由指挥中心的控制台上的平板电脑（移动终端）设备提供设置和控制界面。

## 1.3编写目的

* 本文档是人才市场大数据智能分析展示系统概要设计与详细设计说明书，供项目后期阶段参考使用；
* 本文档可以作为项目验收标准之一；
* 本文档可以作为软件维护的参考资料。

## 1.4预期读者和阅读建议

* 此概要设计与详细设计说明书针对用户、设计人员、开发人员及测试人员。本文分别介绍了产品的详细情况、用户功能及运行环境，系统的功能的具体描述。

## 1.5参考资料

本文档参考资料如下：

* 《第十届中国大学生服务外包创新创业大赛企业赛题手册》
* 《第十届中国大学生服务外包创新创业大赛参赛手册》
* 《大数据标准化白皮书（2018版）》
* 《国家大数据标准手册》
* 《国家信息中心：大数据标准化政策、组织、体系及进展》
* 《Echarts API说明文档》
* 《HighCharts 详细使用及API文档说明》
* 《[Hadoop分布式文件系统:架构和设计](https://www.baidu.com/link?url=PhyFfR4xyR3C7Xy2YX58DkAckZHHPhj35nFz5IMPdi3-J2v3dlAnvjChBvmeX_tDwUOXZunhewJrkiAxk9wio5sZeKcviptFHXFPv1mBdQy&wd=&eqid=f672b7960006170a000000025c7640d7" \t "/Users/nianxilai/Documents\\x/_blank)》
* 《Apache Hive官方文档》
* 《Hbase Book - Apache HBase ™ Reference Guide》
* 《计算机软件文档编制规范GB/T8567-2006》
* 《软件设计文档国家标准详细设计说明书GB8567——88》
* 《软件设计文档国家标准概要设计说明书GB8567——88》

# 需求分析

## 2.1 系统功能性需求分析

由于人力资源市场就业和招聘数据过于繁杂，求职意向和招聘需求信息不透明导致企业和个人获取信息成本过高。为了解决人力资源市场存在的这些问题，我们开发出了一个面向就业和招聘市场的基于大数据分析和数据可视化技术的大屏交互系统。这是一个集动态展示、智能分析、科学预测、个性推荐于一体的大数据研究类项目。本系统可以为求职者、企业和高等院校用户提供清晰的信息展示和全方位的数据参考服务。

根据大数据软件的相关特点，结合人力市场的相关需求，我们分析得出潜在用户的软件需求。我们将设计六大核心模块来满足用户的服务需求，这六个功能模块包括动态展示模块、智能分析模块、科学预测模块、综合推荐模块、移动端控制模块以及数据后台管理模块。动态展示模块需要包括全国总体情况、热门行业情况、热门城市情况以及综合数据情况四个方面，主要显示动态实时数据；智能分析模块需要包括相关信息分析和热门城市情况分析两个方面，主要是将数据在Hadoop大数据生态系统中进行静态分析并展示分析结果；科学预测模块需要将系统数据库中的历史数据通过算法分析并展示预测结果；个性推荐模块需要根据求职者个人实际情况对历史数据进行建模分析得出推荐方案；数据后台管理模块需要为用户提供数据管理和集群管理的相关功能；移动端控制模块需要为用户提供在移动设备上控制数据大屏的功能。

## 2.2 系统非功能性需求分析

### 2.2.1 用户界面设计规则

根据大数据平台软件的特点，设计符合个人、企业、高等院校用户需求的、美观大方的数据展示平台用户界面、数据后台管理平台用户界面和移动端控制软件用户界面。

### 2.2.2 外部接口

* Echarts接口
* HighCharts接口
* BaiduMap API 接口

### 2.2.3 出错处理设计

* 出错处理：在错误发生时，给出出错的原因。
* 出错信息

　　1）按键无效请重试

　　2）设备已经关闭，请检查

* 补救措施

　　1）硬件重新启动

2）软件重新启动

* 说明故障出现后可能采取的变通措施，包括：
* 后备技术：当原始系统数据万一丢失时启用的副本的建立和启动的技术。
* 降效技术：使用另一个效率稍低的方法来求得所需结果的某些部分，如手工操作和数据的人工记录。
* 恢复和再启动技术：使系统从故障点恢复执行或使系统从头开始重新运行的方法。

### 2.2.4 系统维护设计

* 根据用户新的需求添加或删除现有功能
* 根据用户反馈信息进行系统优化

# 概要设计说明

## 3.1设计目标与原则

3.1.1设计目标

由于人力资源市场就业和招聘数据过于繁杂，求职意向和招聘需求信息不透明导致企业和个人获取信息成本过高。为了解决人力资源市场存在的这些问题，我们开发出了一个面向就业和招聘市场的基于大数据分析和数据可视化技术的大屏交互系统。这是一个集动态展示、智能分析、科学预测、个性推荐于一体的大数据研究类项目。本系统可以为求职者和企业提供清晰的信息展示和全方位的数据参考服务。

### 3.1.2 建设原则

* **在建设的原则上，秉承以下的相关原则：**
* **安全性、保密性、稳定性**

人才市场大数据智能分析展示系统中保存的内容是涉及个公司重要信息和个人隐私的数据，且具有大量的历史数据， 因此系统建设首先要考虑的就是系统中数据的安全性与保密性，必须通过多种方式来进行系统数据的加密与验证，从而防止系统数据的泄露，同时系统建设完成后，会与其他业务系统之间进行交互，防止其影响企业内其他的业务，必须要保证系统的稳定。

* **模块化、规范化、标准化**

系统在进行设计的过程中，为提高开发效率，保证各个模块之间的松耦合性，采用了模块化的开发方式，同时，为了保证系统在后期维护过程中，易于维护，便于使用， 在设计与开发过程中，秉承规范化与标准化的思想，确保系统的健壮性。系统应遵循软件开展相关的国际标准和设计规范，应采用标准化接口和组件化设计，同时，还应支持与其它各系统的互联和数据共享交换。

* **实用性、先进性、可持续发展**

系统在设计开发过程中，既要保证系统的先进性，与业界的先进技术接轨，同时也要保证系统设计开发的实用性，保证系统建设完成后真正的可用能用，同时也要充分的考虑到系统未来的发展方向，从可持续发展的角度来进行系统的建设。

* **灵活性、易用性、前瞻性**

系统应能针对系统的特征和功能业务的特点灵活地设计个性化的应用定制，同时还应达到人机界面友好、简单易用、操作方便等要求。同时系统不仅要满足现在的要求，还应当考虑将来发展的需要。

## 3.2 运行环境

### 3.2.1硬件环境

表 3-1 硬件环境

|  |  |
| --- | --- |
| 硬件 | 要求 |
| 数据库服务器 | Intel core i5、内存4G以上、  硬盘500GB、100M网卡 |
| 应用服务器 | Intel core i5、内存4G以上、  硬盘500GB、100M网卡 |
| 无线路由器 | 802.11BGN、无线路由 |
| 客户端PC | 奔腾双核、内存2G以上、  硬盘500GB、100M网卡 |

### 3.2.2软件环境

表 3-2 软件环境

|  |  |
| --- | --- |
| 软件 | 要求 |
| 服务器操作平台 | Windows 10，Ubuntu16.04 |
| 应用服务器 | Tomcat8.5 |
| 网络环境 | Internet |
| 支持协议 | TCP/IP, HTTP, SSH |
| 数据库 | MySOL10.1.7, HBase , hive数据仓库 |
| 支撑环境 | JDK1.7及以上， |
| 开发工具 | Eclipse，Oracle VM，VirtualBox，FileZilla，  Xftp，secureCRT，Android Studio，Genymotion， |

## 3.3 系统总体架构

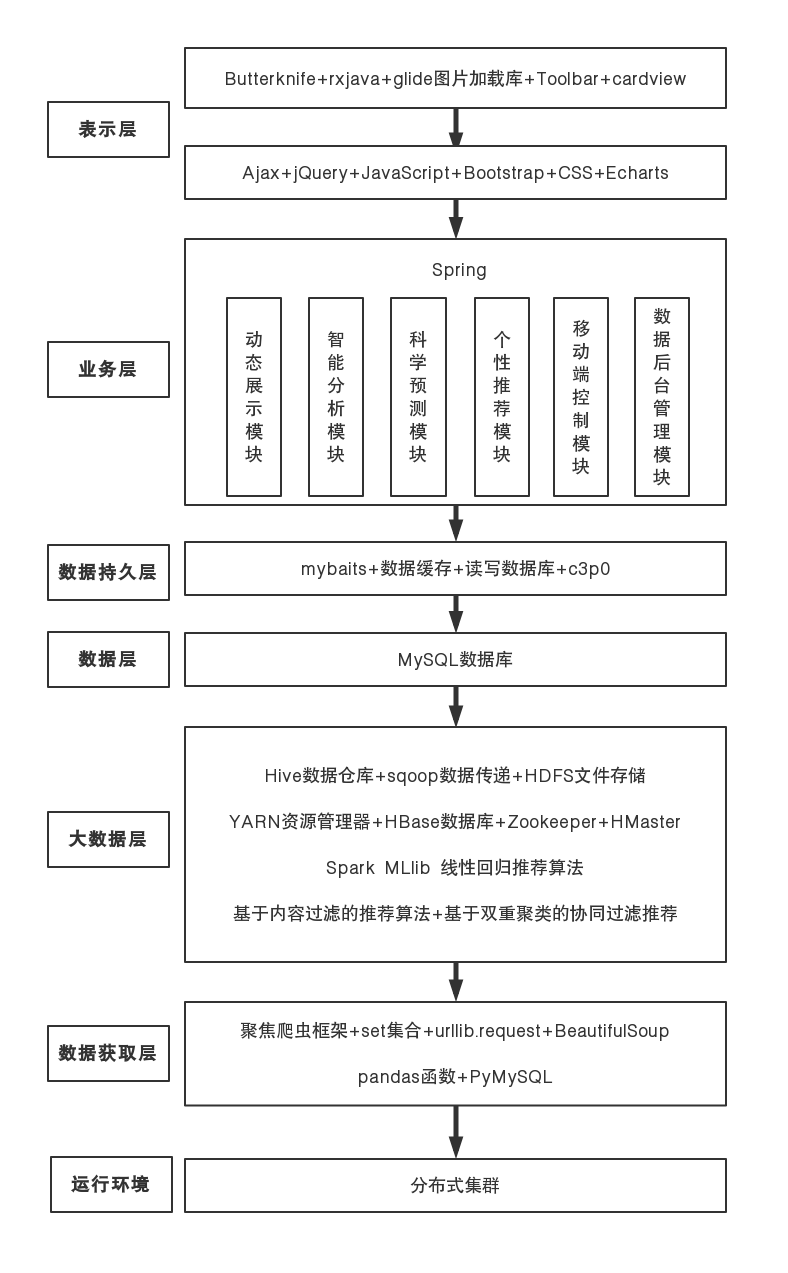


图 3-1 系统总体架构图

首先在数据获取方面，我们为了全面和实时的获取人才需求，选择了聚焦爬虫为本次数据采集的基本架构。聚焦爬虫又称为垂直网络爬虫，它可以理解为一个细化了的通用爬虫，可以对行业、发布时间、招聘内容等诸多因素进行筛选。我们通过设定需求，有选择性地访问人才招聘网页和相关链接，获得最新的人才需求信息。

其次在大数据系统架构方面我们采用基于Hadoop的技术扩展和封装，构建了分布式的运行环境，进行高效的运算和分析。围绕Hadoop衍生出相关的大数据技术，应对传统关系型数据库较难处理的数据和场景，例如针对非结构化数据的存储和计算等，充分利用Hadoop开源的优势，伴随相关技术的不断进步，其应用场景也将逐步扩大，目前最为典型的应用场景就是通过扩展和封装Hadoop来实现对互联网大数据存储、分析的支撑。

然后在应用服务器的选择上，统一人才市场大数据智能分析展示系统其并发量与访问量相对可控，考虑到性价比的因素，选择了免费开源的 Tomcat 应用服务器，因为应用程序的部署很简单，你只需将你的WAR放到Tomcat的webapp目录下，Tomcat会自动检测到这个文件，并将其解压。由于Java的跨平台特性，基于Java 的Tomcat也具有跨平台性。Tomcat 也可以与其它一些软件集成起来实现更多的功能。

最后在数据库的选择上，我们选择了MySQL数据库。MySQL是一个高性能且相对简单的数据库系统，与一些更大系统的设置和管理相比，其复杂程度较低。而且MySQL对多数个人用户来说是免费的，MySQL数据库没有用户数的限制，多个客户机可同时使用同一个数据库，可利用几个输入查询并查看结果的界面来交互式地访问MySQL。最重要的是MySQL是目前能得到的最快的数据库，这对于我们做大数据系统来说非常重要。

## 3.4 系统功能设计

本系统共有6个功能模块，包括动态展示模块、智能分析模块、科学预测模块、综合推荐模块、移动端控制模块以及数据后台管理模块。

### 3.4.1 动态展示模块

* **动态展示模块包括全国总体情况、热门行业情况、热门城市情况以及综合数据情况四个页面。本模块将获取的数据按照不同的特点进行分类并在不同的图表中显示动态实时数据。**
* **全国总体情况页面实时动态展示全国各大主要城市和主要行业的人力资源招聘的总体情况。**

①行业招聘热度地图

②全国城市招聘需求图

③全行业工资状况图

* **热门行业情况页面实时动态展示热门行业的人力资源招聘的总体情况。**

①行业招聘热度图

②行业招聘需求比例图

③热门行业工资状况图

* **热门城市情况页面实时动态展示热门城市的人力资源招聘的总体情况。**

①北京市实时热门行业需求

②北京实时招聘资讯

③北京实时热门行业工资状况图

④北京实时人员学历招聘需

* **综合数据情况页面实时动态展示重要字段之间相关性图表。**

①国企、民企工资区间人数状况图

②工资与学历状况图

③工作经验需求状况图

④工作经验与工资状况图

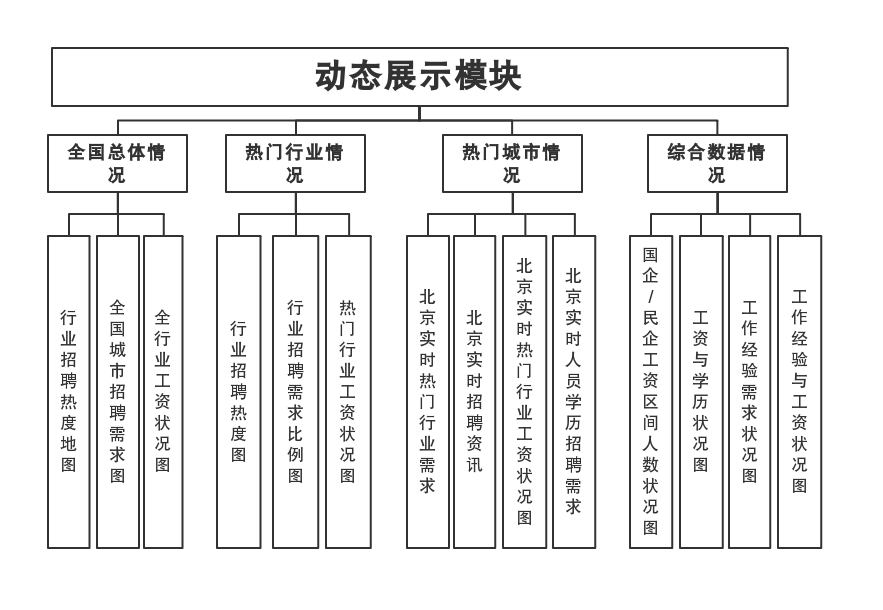
****

图 3-2 动态展示模块功能结构图

### 3.4.2 智能分析模块

* **智能分析模块包括相关信息分析和热门城市情况分析两个页面。本模块将获取的数据在Hadoop大数据生态中进行静态分析并在不同的图表中展示分析结果。**
* **相关信息分析页面从需求、待遇、学历和工资四个维度对相关信息进行分析并展示分析结果。**

①热门行业招聘需求

②热门行业待遇等级

③各学历人数占比

④学历与工资关系图

* **热门城市情况分析页面通过分析热门城市的相关数据并展示分析结果。**

①热门行业招聘需求

②热门城市工资情况图

③热门城市招聘数据

④部分城市工作满意度

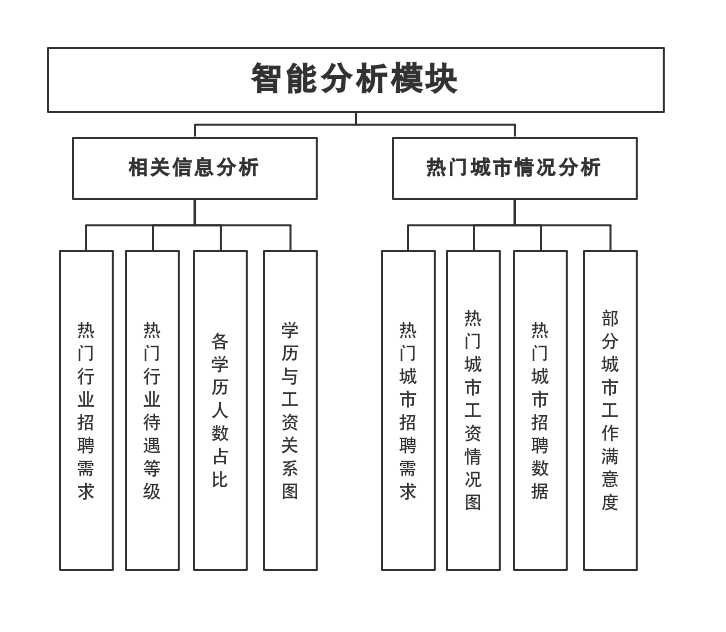


图 3-3 智能分析模块功能结构图

### 3.4.3 科学预测模块

* **科学预测模块将系统数据库中的历史数据通过Hadoop大数据生态进行算法计算并通过不同的图表展示预测结果。**
* **趋势预测页面：运用预测算法进行趋势预测。**

①二零一九年热门城市就业推荐指数

②二零一九年行业信息预测表

③二零一九年企业各渠道招聘比例预测

④二零一九年各学历层次工资预测

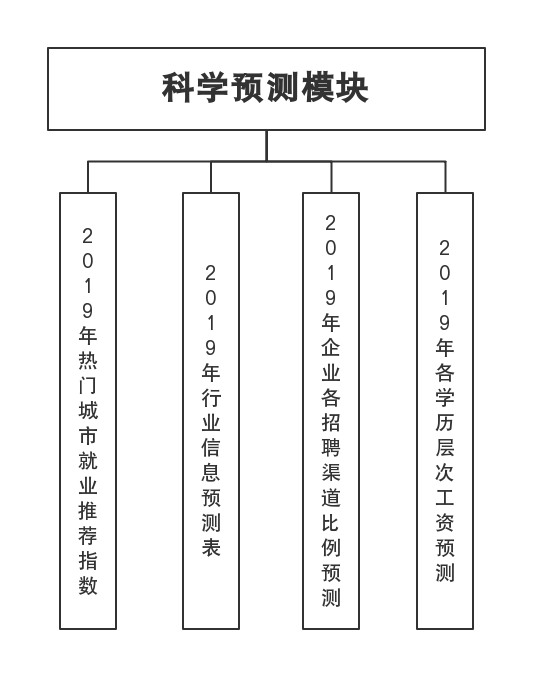


图 3-4 科学预测模块功能结构图

### 3.4.4 个性推荐模块

* **个性推荐模块通过根据求职者个人实际情况对历史数据进行建模分析得出推荐方案。**

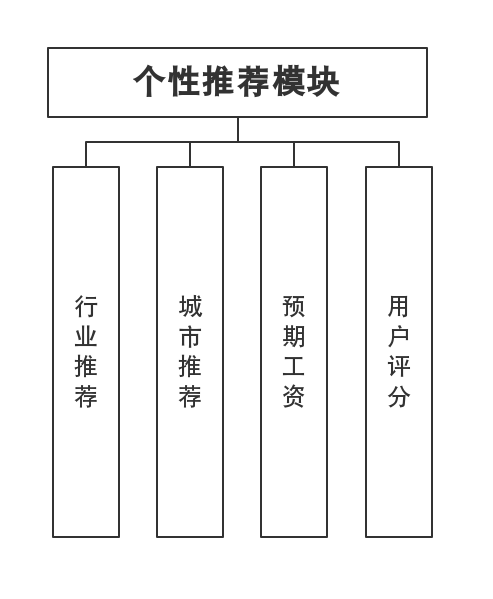


图 3-5 个性分析模块功能结构图

### 3.4.5 数据后台管理模块

* 通过这个系统可以进行数据的加载更新、管理HDFS文件系统、监控集群节点状态、监控集群数据状态、进行用户管理。同时，通过数据后台管理系统，我们可以快速的切换数据源和配置相关图表，以实现系统的通用性和可扩展性。

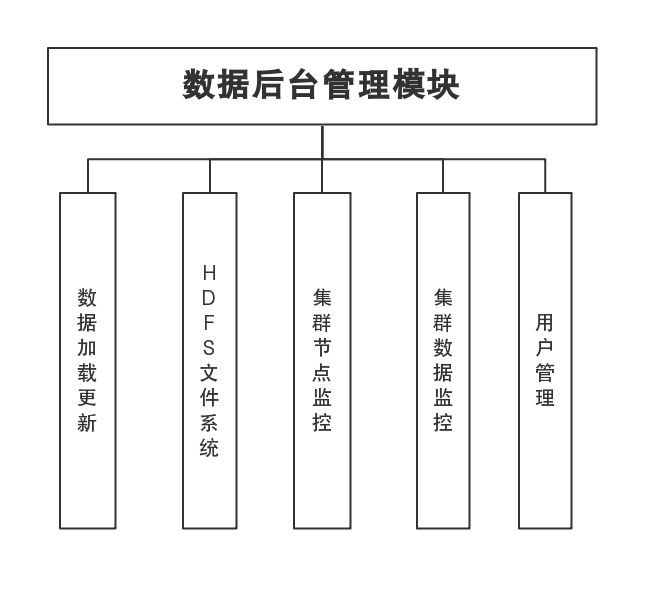


图 3-6 数据后台管理模块功能结构图

### 3.4.6 移动端控制模块

* 移动端控制模块可以通过智能手机和平板电脑对数据大屏实现远程控制。

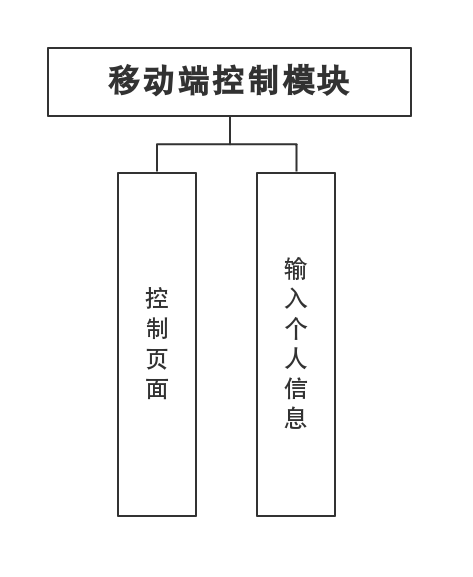


图 3-7 移动端控制模块功能结构图

# 第四章 技术路径综述

## 4.1系统总体流程

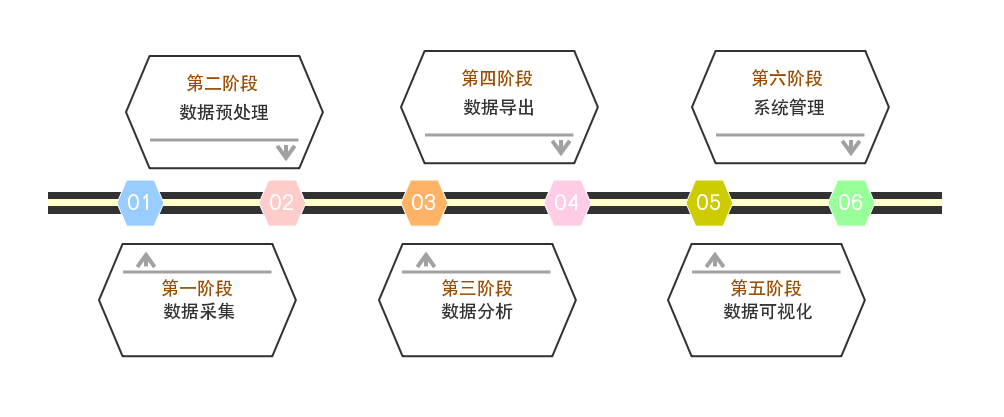


图 4-1 系统总体流程图

* **本项目的系统流程主要分为六个步骤。**
* 第一阶段是采用聚焦爬虫的基本架构，通过一系列的登录验证、URL管理、页面获取、页面解析、数据清洗、数据存储等过程来获取招聘信息;
* 第二阶段是采用Hive数据仓库对爬取到的数据进行探索分析，然后用数据清洗和属性规约等方法对数据进行处理，最后得到可以使用的数据；
* 第三阶段是使用Hive数据仓库进行离线数据分析，HBase分布式数据进行实时数据的计算分析，Spark MLlib算法进行信息预测，利用基于内容过滤和双重聚类的协同过滤推荐算法来进行个性化的职业推荐;
* 第四阶段是对模型的部署和应用，通过基于Hadoop Sqoop数据迁移工具将模型运算结果输出到MySQL数据库。
* 第五阶段是通过Ajax异步数据获取的方法将数据展现到ECharts图表中，最终将结果清晰直观的呈现到页面中.
* 第六阶段是通过开发数据后台管理系统和移动Android端实现对数据大屏数据管理以及集群管理和远程控制功能。

## 4.2第一阶段：数据采集（基于聚集爬虫的数据采集）

### 4.2.1聚焦爬虫工作过程

为了全面和实时的获取人才需求，我们选择了聚焦爬虫为本次数据采集的基本架构。聚焦爬虫又称为垂直网络爬虫，它可以理解为一个细化了的通用爬虫，可以对行业、发布时间、招聘内容等诸多因素进行筛选。我们通过设定需求，有选择性地访问人才招聘网页和相关链接，获得最新的人才需求信息。聚焦爬虫的工作流程如图4-2。

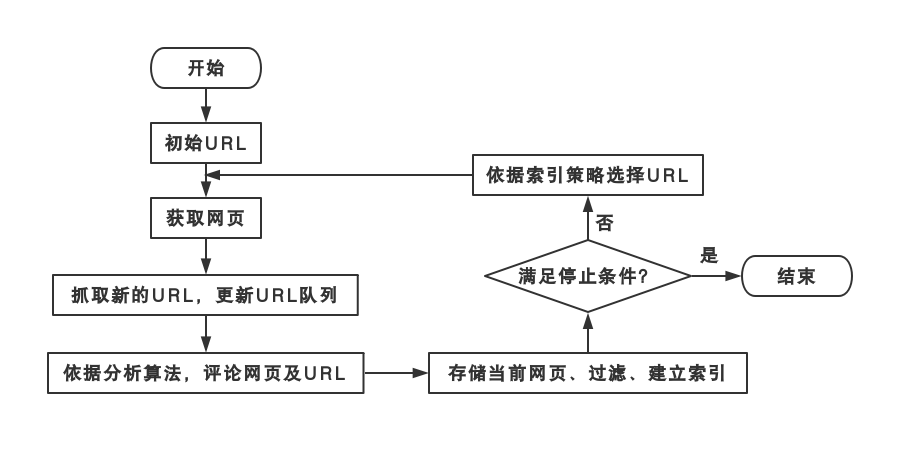


图 4-2 聚焦爬虫工作流程

### 4.2.2 登录验证

在招聘信息爬取过程中，大部分网站是要求用户登录的，未登录的用户就无法获得招聘信息。这就需要在数据采集过程中自动登录和保持登录。我们设计并实现了登录验证模块，其主要功能就是自动登录和保持登录。利用模拟的登录请求信息进行登录后，服务器会自动分配一个身份凭证sessionID，但是一般情况下sessionID会在服务器设定的时间（30分钟）后失效。这个时候服务器就会要求用户重新登录，如果用户没有响应，服务器就会跳转到403Forbidden错误页面，这也就以为这登录状态失效。为了应对登录状态失效以及页面异常401unauthorized问题，我们借助cookieJar对象来获取cookie信息，通过这一信息每隔一段时间重新模拟登录，以此来保持登录状态。

### 4.2.3 URL管理

我们利用Python来编写爬虫，Python环境中的set集合和其他语言中的set类似，它是一个无序不重复元素集，所以set支持关系测试和消除重复元素。为了防止重复抓取和循环抓取，特别是两个URL互相指向时爬虫会进入死循环。使用Python的set集合对URL进行管理就可以很好的避免这一情况的发生。具体的URL管理流程如图4-3。

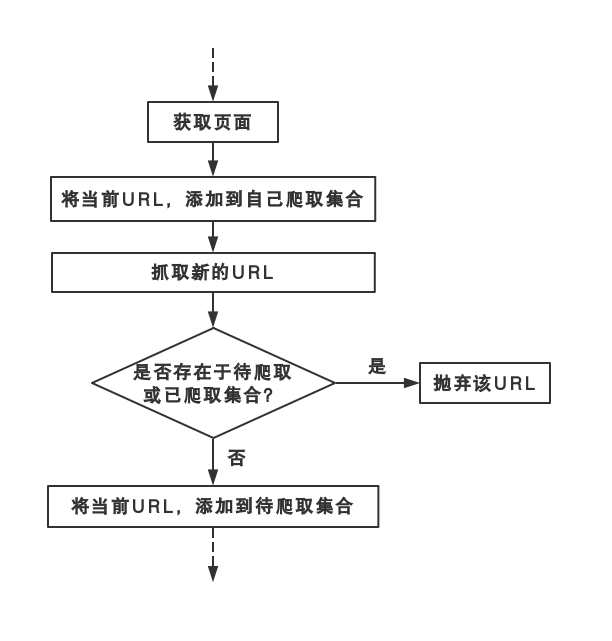


图 4-3 URL管理流程

### 4.2.4 页面获取

Python3中有自带的urllib.request模块，我们借助这一模块完成页面下载任务，在实现过程中，定义页面下载类 HtmlDownLoader，及其方法downloade() 返回页面内容，其具体的代码如下：



图 4-4 页面获取代码

### 4.2.5 页面解析

页面解析机制是将downloade()所返回的数据进行解析，便于聚焦计算和存储。网页解析的方法依据网页内容构成的不同，分为模糊匹配和结构化解析两种。 由于招聘网站通常是结构化的，所以结构化解析更适用。我们使用的是Python3中自带的BeautifulSoup库构建HTML页面解析器。BeautifulSoup库将HTML信息拆解为对象处理，全部转换为字典或数组。这种方法与使用正则表达式的模糊匹配解析器相比，结构化解析节约了正则学习的高成本。具体的代码如下所示， 将一段 HTML 代码（html\_doc）传入 BeautifulSoup 的构造方法，就能得到一个文档的对象（soup）。 然后使用findAll（）方法即可获取网页中有价值的信息。

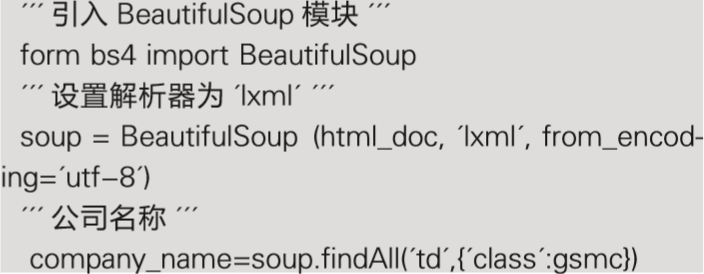


图 4-5 代码

### 4.2.6 数据清洗

我们使用Python编写的pandas函数对以获取的数据进行数据清理，从而将冗余的重复的无用的信息排查出去，并且对数据进行分类整理。具体实现代码如下所示

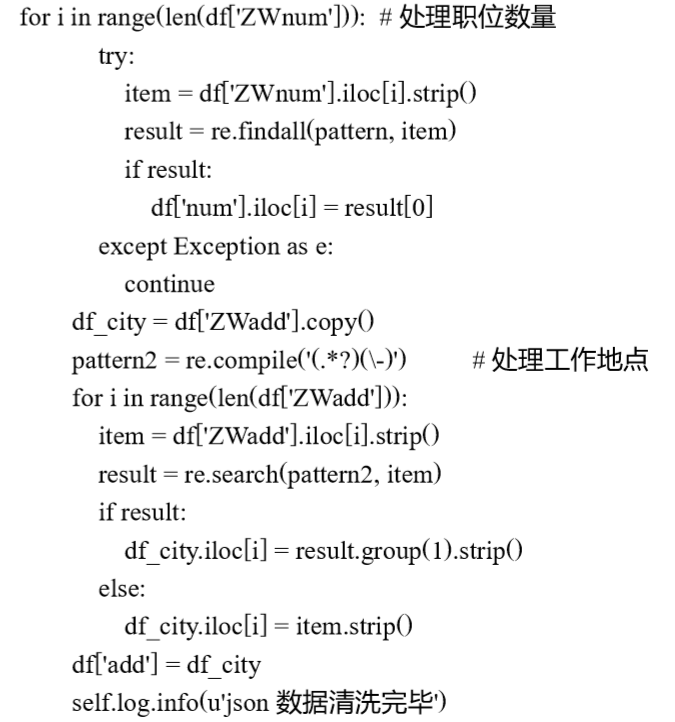


图 4-6 pandas函数代码

### 4.2.7 数据存储

在Python3环境下使用PyMySQL操作将招聘信息存储于MySQL 数据库。具体的存储步骤如下：

1. 引入 pymysql.cursors模块，构建字典进行连接参数的管理。
2. 连接到数据库，首次爬取该站点使用 “create table” 创建数据表。
3. 若非第一次爬取该站点则使用“insert”语句将页面解析所获取内容存储到数据表。
4. 将此次插入操作写入爬取日志。

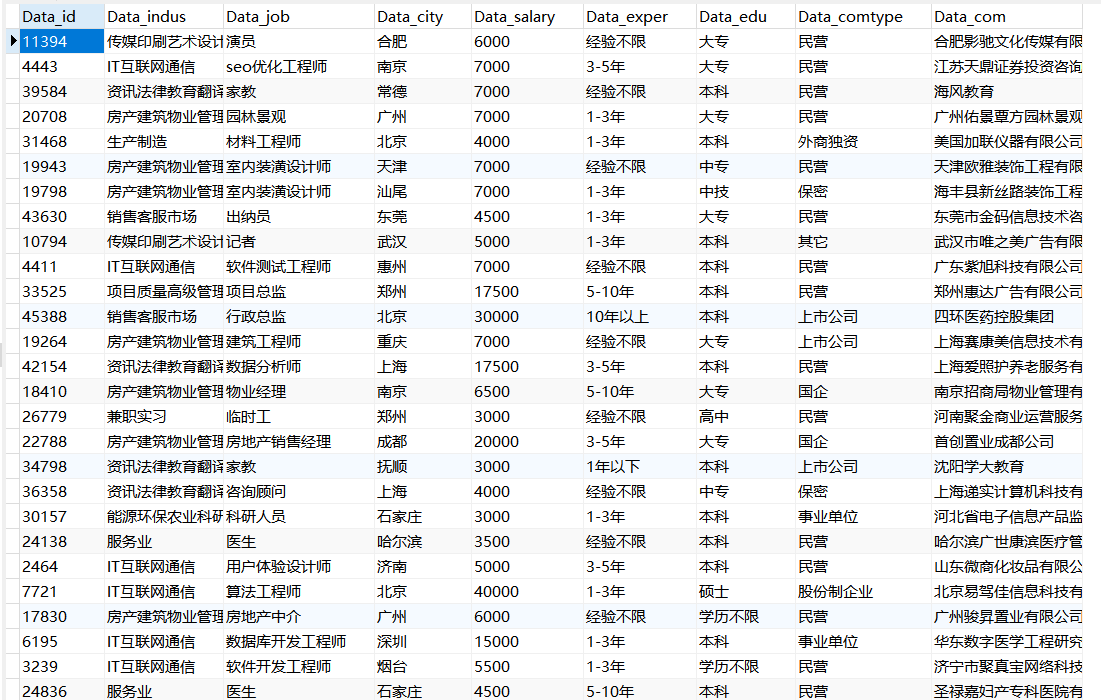


图4-7 数据库数据

### 4.2.8 补充说明

对目标站点连续访问不同网页，如果不限制爬虫的请求频率，爬虫的效率只会受到所处终端的处理能力和带宽的限制，因此爬虫的访问频率会非常高。通过增加线程的休眠时间，降低访问频率，实现模仿人为浏览的行为避免被反爬虫机制侦测到。具体代码如下:

import time

Time.sleep( 0.5)

## 4.3第二阶段：数据预处理（基于Hive的数据预处理）

### 4.3.1 基于Hive的数据预处理过程

我们项目对前期数据分析处理部分主要包括以下几个步骤：

1）从已有的招聘网站数据源进行数据抽取；

2）对抽取的数据进行数据探索分析，包括数据缺失值与异常值的探索分析；

3）根据数据探索分析结果，对数据进行预处理，包括数据清洗，属性规约，进而得到预测建模和个性推荐建模所需要的数据。

建模所需数据将薪资S、企业类型C、工作经验E、行业I 、工作岗位J 、学历E、工作城市C七个指标作为招聘信息价值指标，记为SCEIJEC模型指标数据。

### 4.3.2 数据抽取

以2019年3月10日为结束时间，选取宽度为4个月的时间段作为分析观测窗口（观测窗口的意思是，以过去某个时间点为结束时间，某一时间长度作为宽度，得到历史时间范围内的一个时间段），共计1043525条记录。记录包含了行业、岗位、企业名、工作地点、薪资、要求学历、企业类型等十几个属性。

### 4.3.3 数据探索分析

使用Hive数据仓库对数据进行缺失值和异常值分析，得出数据的规律以及异常值。根据爬虫获取的数据进行数据探索分析，分析目标如下：

1、原始数据工作经验、学历、薪资和工作地点中存在空值，可能是由于招聘信息不规范或者爬取的时候数据丢失造成的。

2、原始数据薪资中存在极其不合理数据例如新疆招聘博士后工资小于1000等情况，可能是由于招聘网页显示错误等原因造成的。

### 4.3.4 数据预处理

在使用Hive针对所有数据进行初步探索后，就需要采用数据清洗、属性规约等方法对数据进行进一步的处理。

**1、数据清洗**

通过数据探索分析发现数据中存在缺失值、属性值为空或者薪资极其不合理的数据等。由于原始数据量庞大，这类数据占比较小，对结果影响不大，因此对其进行丢弃处理。具体数据清洗规则如下：

1）丢弃STWIJEC模型中属性为空的记录；

2）丢弃工资极其不合理的情况，删除合理区间外的数据；

3）将经过数据清洗后的数据通过数据迁移工具Sqoop存储到Hive数据仓库中。

**2、属性规约**

从清洗后的数据中选取与STWIJEC模型指标相关的属性。

图4-5 为属性规约后的数据集：



图 4-8 数据集

## 4.4第三阶段：数据分析①(基于Hive的离线数据分析)

### 4.4.1 Hive进行离线数据分析的优势

引入Hadoop框架处理数据，我们首先考虑使用Map/Reduce函数来实现离线数据的分析，但是考虑到我们需要耗费时间和精力去编写具体的Map/Reduce函数，这种函数较为底层，不仅开发效率低，而且不易维护和扩展。Hadoop生态圈中的Hive组件可解决这一问题，Hive进行静态数据分析有以下几点优势：

* Hive最大的特点是通过类SQL来分析大数据，而避免了写MapReduce程序来分析数据，这样使得分析数据更容易。
* 数据存储方面：由于Hive的底层存储架构是 HDFS，而 HDFS是适合大数据集的存储和计算的，而且 Hadoop的底层计算框架是 MapReduce，因此 Hive 是一个非常适合大数据集的存储和计算的。
* 数据处理方面：因为Hive语句最终会生成MapReduce任务去计算，所以不适用于实时计算的场景，它适用于离线分析。
* 高扩展性：Hive 的 SQL 语句经过引擎的编译在底层都会转换为一个个的MapReduce 任务，因此 Hive 和 MapReduce 一样都具有高扩展的特性。

由于Hive的高扩展性、高容错、存储数据量大等特点使得其特别适合大数据处理和分析，底层存储利用 HDFS 的架构，有利于非结构化数据的存储，无论是在存储数据量和批量分析计算和传统的关系型数据库相比都具有更多的优势。因此在离线数据这一块我们系统采用Hive进行。

### 4.4.2 Hive数据仓库基本架构

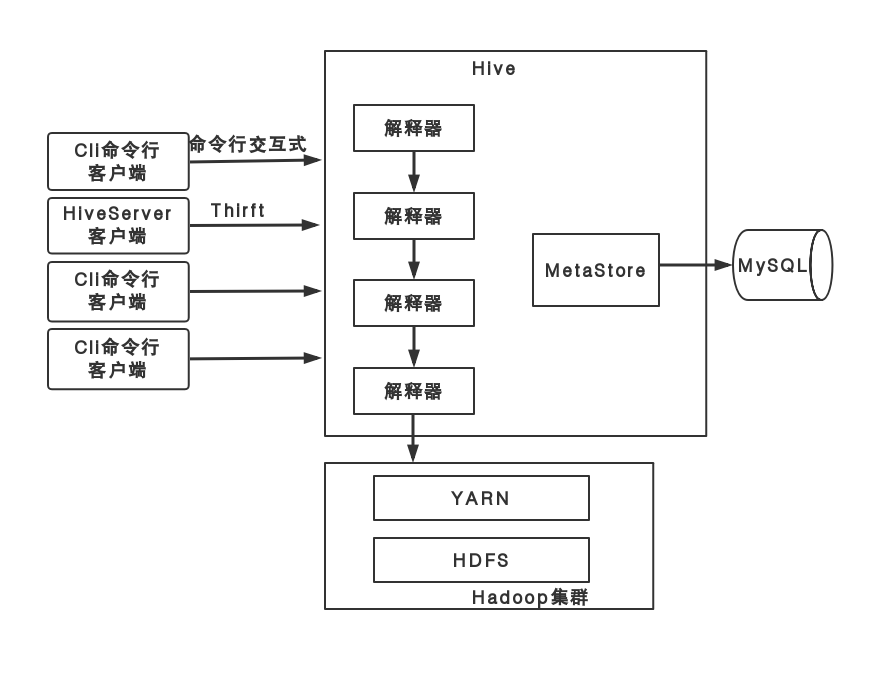


图 4-9 Hive数据仓库架构架构图

由Hive数据仓库的架构图我们可以知道Hive的核心驱动引擎，而驱动引擎由四部分构成，它们的功能分别是：

* 解释器：解释器的作用是将HiveSQL语句转换为语法树（AST）。
* 编译器：编译器是将语法树编译为逻辑执行计划。
* 优化器：优化器是对逻辑执行计划进行优化。
* 执行器：执行器是调用底层的运行框架执行逻辑执行计划。

Hive底层存储：Hive的数据是存储在HDFS上的，Hive中的库和表可以看做是对HDFS上的数据做的一个映射。所以HVIE必须运行在一个Hadoop的集群上的。

Hive语句执行过程：Hive中的执行器是将最终要执行的MapReduce程序放到YARN上以一系列Job的方式去执行。

Hive的元数据存储：Hive的元数据是一般是存储在MySQL这种关系型数据库上的，Hive和MySQL之间通过MetaStore服务交互。

### 4.4.3 离线数据分析设计概要

离线数据的分析模块主要完成 T+1 数据指标的统计分析和关联分析，如热门行业招聘需求分析、热门行业待遇等级分析、各学历人数占比分析、学历与工资关系的分析、热门城市招聘需求分析、热门城市工资情况分析和部分城市工作满意度的分析。通常一个任务都会包含几个子任务，此时一个主进程和多个子进程一起共同完成整个复杂的任务。本系统主要利用 MapReduce 完成离线数据的分析工作。同时采用了 HiveSQL 语言分析数据，HiveSQL 自动的会转化为 MapReduce 程序去处理。

离线数据分析是将Hive数据库中进行过预处理的数据导入到分区表中，以时间为分区字段，Hive分区为静态分区，HiveSQL做离线批处理，将查询结果存放在HDFS分布式文件存储系统中。数据导出时通过数据迁移工具Sqoop将数据导入到MySQL数据库以供前端展示时调用。

## 4.5第三阶段：数据分析②(基于HBase的实时数据分析)

### 4.5.1 Hbase技术介绍

由于Hadoop是一个高容错、高延时的分布式文件系统和高并发的批处理系统，不适用于提供实时计算，而HBase是一种构建在HDFS之上的**分布式、面向列**的存储系统。在需要实时读写、随机访问超大规模数据集时，可以使用HBase，所以本系统采用基于Hadoop基础上的HBase分布式数据来在进行实时数据的计算分析。

### 4.5.2 HBase系统架构

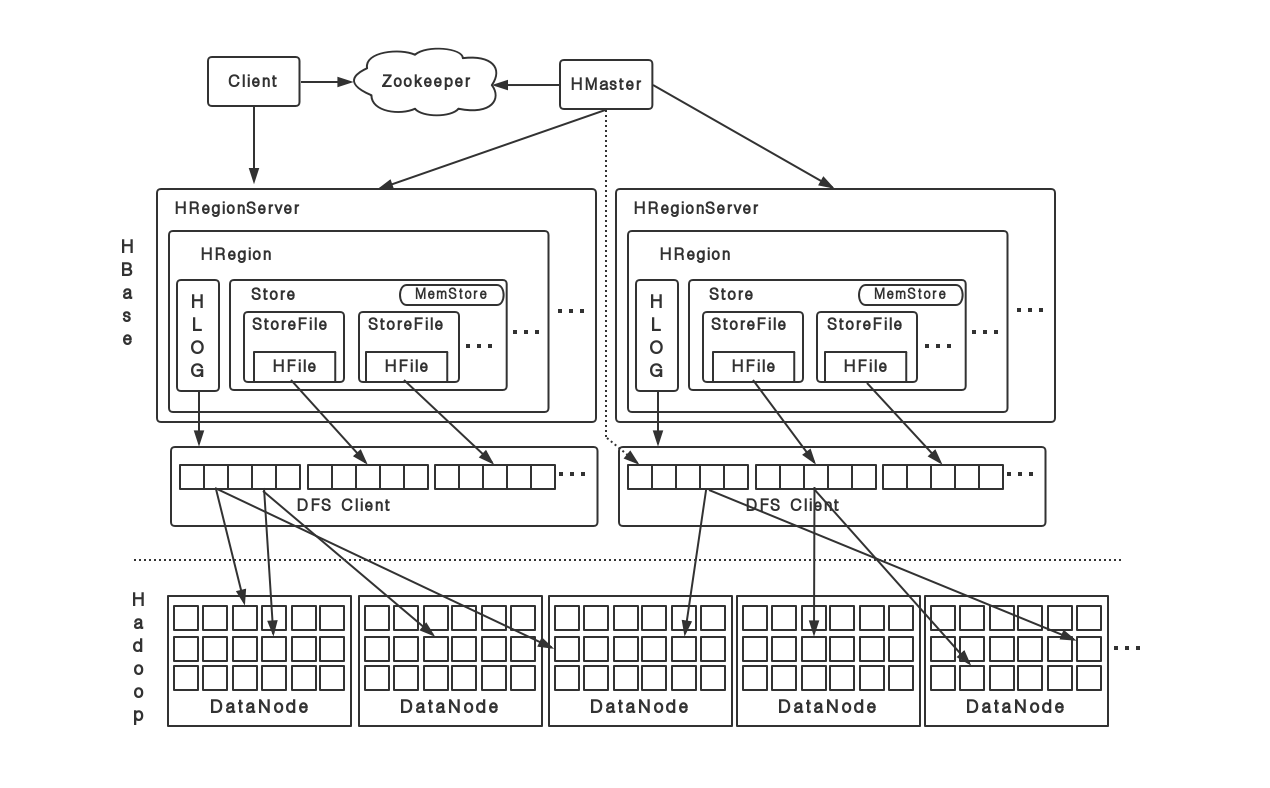


图 4-10 HBase系统架构图

从HBase的架构图上可以看出，HBase中的组件包括Client、Zookeeper、HMaster、HRegionServer、HRegion、Store、MemStore、StoreFile、HFile、HLog等。

HBase中的每张表都通过行键按照一定的范围被分割成多个子表（HRegion），默认一个HRegion超过256M就要被分割成两个，这个过程由HRegionServer管理，而HRegion的分配由HMaster管理。

从根本上说，HBase能提供实时计算服务主要原因是由其架构和底层的数据结构决定的，即由LSM-Tree + HTable(region分区) + Cache决定——展示端端可以直接定位到要查数据所在的HRegion server服务器，然后直接在服务器的一个region上查找要匹配的数据。

### 4.5.3 HBase读取数据流程

1）Client先访问Zookeeper，从meta表读取region的位置，然后读取meta表中的数据。meta中又存储了招聘信息表（如下表xxx）的region信息；

2）根据namespace、表名和rowkey在meta表中找到对应的region信息；

3）找到这个Region对应的Regionserver；

4）查找对应的region；

5）先从MemStore找数据，如果没有，再到BlockCache里面读；

6）BlockCache还没有，再到StoreFile上读(为了读取的效率)；

7）如果是从StoreFile里面读取的数据，不是直接返回给客户端，而是先写入BlockCache，再返回给客户端。

表 4-1 招聘信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 主键/列族 | 字段名称 | 字段含义 | 字段值举例 | 备注 |
| Rowkey | —— | 表主键（id） | 1000 |  |
| Timestamp | —— | 时间戳 | 1414939140000 | Long型 |
| Info | —— | 列族 | —— | 招聘信息 |
| Data\_indus | 职业方向 | 房地产 |  |
| data\_job | 具体职业 | 系统架构师 |  |
| Data\_city | 所在城市 | 北京 |  |
| Data\_salary | 薪资 | 5000 |  |
| Data\_exper | 工作经验 | 3年 |  |
| Data\_edu | 学历 | 本科 |  |
| Data\_comptyp | 工作性质 | 国企 |  |

### 4.5.4 数据Flush过程

1）当MemStore数据达到阈值（默认是128M，老版本是64M），将数据刷到硬盘，将内存中的数据删除，同时删除HLog中的历史数据；

2）并将数据存储到HDFS中；

3）在HLog中做标记点。

## 4.6第三阶段：数据分析③(基于Spark MLlib算法的信息预测)

### 4.6.1 Spark MLlib机器学习库介绍

MLlib是Spark平台中的机器学习库，它让机器学习更加容易使用、可扩展性更强，目的是让开发人员对机器学习更容易入门。MLlib在机器学习上拥有很多优势，大致可以总结为：

1. MLlib基于内存迭代的特点使得它可以完美的适应机器学习算法频繁的使用CPU 和输入输出设备进行多次迭代直到达到足够大的准确率后才停止的特性。
2. MLlib 有 Spark 生态圈作为支撑，它是在 Spark 基础上建立起来的。
3. MLlib提供了多种常用编程语言的 API，方便用户灵活选用自己擅长的语言进行 Spark 机器学习应用的开发。
4. MLlib 的底层基于数值计算库 Breeze 和基础线性代数库 BLAS（Basic Linear Algebra Subprograms）实现，并实现了丰富的工具。
5. MLlib提供了包括特征提取、特征变换以及特征选择等特征工程相关工具以及线性代数、统计学等数学实用工具。

MLlib实现了自己的 Vector（向量）类，并封装了 Breeze 中的向量方法，在 MLlib函数中的参数传递都是使用自己实现的 Vector，Vector 内部有 ToBreeze 方法，可以将MLlib 的 Vector 类转成 Breeze 的向量。Matrix（矩阵）类的实现也是类似的。

MLlib 进行机器学习 API 开发主要基于 DataFrame、Dataset 两种数据结构。在 Spark

中，DataFrame 是一个命名列数据组织的分布式集合，它在概念上相当于关系型数据库中的一个表或者 R、Python 语言中的 data frame 数据结构，但在内核中进行了更丰富的优化，DataFrame 构建有广泛的数据源：可以是现有的 RDD，也可以是 Hive 中的表，还可以是结构化数据文件或外部数据库。

Dataset 是一种可以使用功能或关系操作并行转换的特定域对象的强类型集合，Dataset 可以利用 JVM 对象创建并支持 map、filter 等一系列的转换操作，每个 Dataset 都有一个作为数据集的行的 DataFrame 非类型化视图[16]。随着 Spark 2.0 于 2016 年 5 月发布，Dataset 逐渐成为开发的主流，慢慢取代 DataFrame，当然这个取代的意思只是在Spark2.0 中 Dataset 实现了 DataFrame 已有的 API。

### 4.6.2 线性回归算法原理分析

回归分析是一种统计工具，它利用两个或两个以上变量之间的关系，由一个或几个变量来预测另一个变量。当自变量只有一个的时候，叫做一元线性回归，当自变量有多个的时候，叫做多元线性回归。

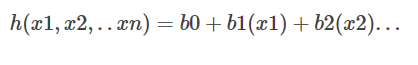
* **概要步骤**

线性回归分析的整个过程可以简单描述为如下三个步骤：

1. 寻找合适的预测函数，即上文中的h(x)，用来预测输入数据的判断结果。这个过程时非常关键的，需要对数据有一定的了解或分析，知道或者猜测预测函数的“大概”形式，比如是线性函数还是非线性函数，若是非线性的则无法用线性回归来得出高质量的结果。
2. 构造一个Loss函数（损失函数），该函数表示预测的输出h(x)与训练数据标签之间的偏差，可以是二者之间的差（h-y）或者是其他的形式（如平方差开方）。综合考虑所有训练数据的“损失”，将Loss求和或者求平均，记为J(θ)函数，表示所有训练数据预测值与实际类别的偏差。
3. 显然，J(θ)函数的值越小表示预测函数越准确（即h函数越准确），所以这一步需要做的是找到J(θ)函数的最小值。找函数的最小值有不同的方法，Spark中采用的是梯度下降法（stochastic gradient descent, SGD)。

* **详细步骤**

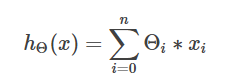
我们这里采用多元线性回归分析，采用的预测函数如下：



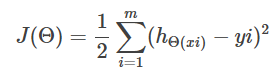
一般情况下，通过对于已知数据的拟合，我们可以得到一个线性方程：



我们统一可以将方程写成如下的格式：



对于θ参数的求解，我们需要去求解θ是否是最优。可以定义一个损失函数，这个损失函数定义为：

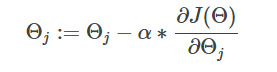


可以通过最小二乘法 和梯度下降法来调节，使得J(θ)有最小值。

### 4.6.3 关于梯度下降算法的选择

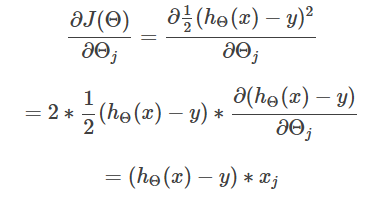
* **批量梯度下降算法**

如上面的公式描述的那样，问题转换为求解J(θ)的极小值问题。由于求解的是极小值，因此梯度方向是偏导数的反方向。



其中α表示的是学习速率。当α过大时，有可能无法收敛。α过小的时候，收敛的速度可能很慢。

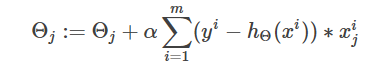
当数据集中只有一个样本的时候，即m=1的时候:



所以梯度下降算法的公式变可以改变为：

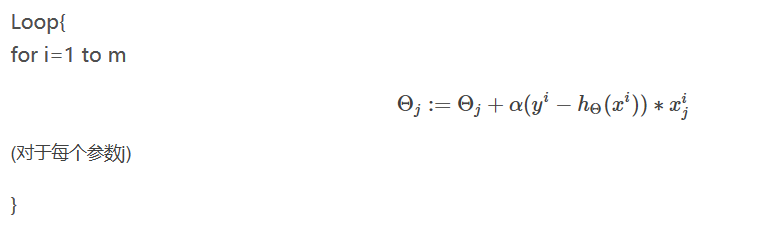


当样本数量不为1的时候，上述公式变为:



这个公式计算每次迭代计算θ时候，需要使用到整个样本数据集，复杂度很高。

* **随机梯度下降算法**



随机梯度下降算法每次读取一条样本，迭代对θ进行更新。然后判断其是否收敛。没有收敛的话，则继续读取样本进行处理。当所有样本都读取完了之后，则[从头开始](https://www.baidu.com/s?wd=%E4%BB%8E%E5%A4%B4%E5%BC%80%E5%A7%8B&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "https://blog.csdn.net/stevekangpei/article/details/_blank)循环。与批量梯度下降算法相比，随机梯度下降算法耗时更少，尤其是当数据量很大的时候。

* **结论**

由于我们数据库中数据量已经达到一百万条，与批量梯度下降算法相比，随机梯度下降算法耗时更少，并且随机梯度算法更好实现，所以最终我们选择随机梯度算法。

### 4.6.4 线性回归算法LinearRegressionWithSGD

* **第一步：在使用LinearRegressionWithSGD算法之前首先将输入数据解析成包括类标和特征向量的LabeledPoint对象的RDD弹性分布式数据集合。**
* 数据集的描述

此数据是根据一系列的特征预测**2019年热门城市就业推荐指数、2019年行业信息、2019年企业各招聘渠道比例、2019年各学历层次工资**，特征类型如下：

表 4-2 数据的特征

|  |  |
| --- | --- |
| **代码** | **名称** |
| Data\_id | 记录ID |
| Data\_indus | 行业 |
| Data\_job | 职业 |
| Data\_city | 工作城市 |
| Data\_salary | 工资（月薪） |
| Data\_exper | 工作经验 |
| Data\_comtype | 公司类型，如民营、国营 |
| Data\_com | 公司名称 |
| Data\_time | 工作时间 |
| Data\_edu | 学历，如专科、本科、研究生和博士生 |
| Number | 招聘人数 |

* **第二步：调用LinearRegressionWithSGD伴生对象中的train方法传输第一步创建的RDD集合和最大迭代次数，在train中主要实现创建一个新的LinearRegressionWithSGD对象。代码如下：**



图 4-11 代码

其中input表示为训练样本，格式RDD(label, features)。numIterations表示迭代次数， stepSize表示迭代步长。miniBatchFraction表示每次迭代的参与的样本的比例。initialWeights表示的是初始的权重。返回值是一个线性回归的模型类。

* **第三步：调用伴生对象的run方法后，由于这个类里面并没有run方法，所以运行的是父类GeneralizedLinearAlgorithm中的run方法，其中的run方法在广义线性回归的类里面。**



图 4-12 代码

* **第四步：进行特征维度的检测，然后检测数据是否缓存，是否降维和是否需要增加偏置项，最后初始化权重。然后调用了optimizer的optimize方法来进行计算。最后调用createModel方法，返回结果。部分代码如下：**



图 4-13 代码

* **第五步：调用GradientDescent类的optimize方法，其内部又调用了一个runMiniBatchSGD方法，它返回的结果是权重。runMiniBatchSGD这个方法是整个线性回归最核心的方法，它思路是：首先初始化好初始权重参数和历史迭代误差的可变数组，然后在每次迭代的时候，广播这个更新的权重到每个RDD。调用treeAggregate算子，每次对数据进行随机采样(无放回采样)，然后先对每个分区的数据进行计算梯度值和误差值，然后接下来对每个分区的计算好的梯度值和误差值进行累加，最后更新权重值。部分代码如下：**

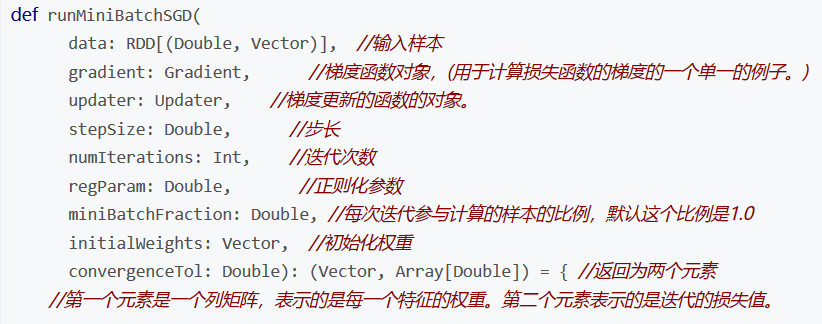


图 4-14 代码

* **第六步：调用LeastSquaresGradient类里面的compute方法来计算。首先是计算 y-h(x)，这个表示的是模拟值和实际值的误差，用diff表示。接下来计算cumGradient= x\*((y-h(x)))。这些计算公式和之前推导的公式都是一致的，最后返回损失值。部分代码如下：**

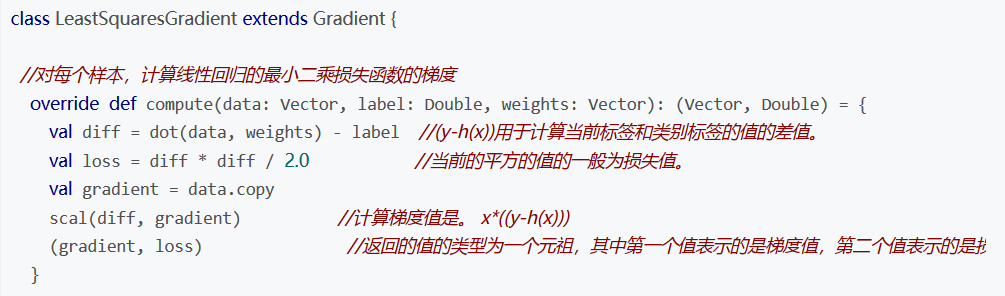


图 4-15 代码

* **第七步：在计算完之后需要对权重进行更新,更新使用的是SimpleUpdater。首先会计算alpha值(学习速率)。alpha值在刚开始的时候更新比较快，后面更新比较慢。thisIterStepSize表示的是alpha值，平方根的倒数作为alpha可以确保在迭代初期的时候迭代的比较快，在迭代的后期比较慢。部分代码如下：**

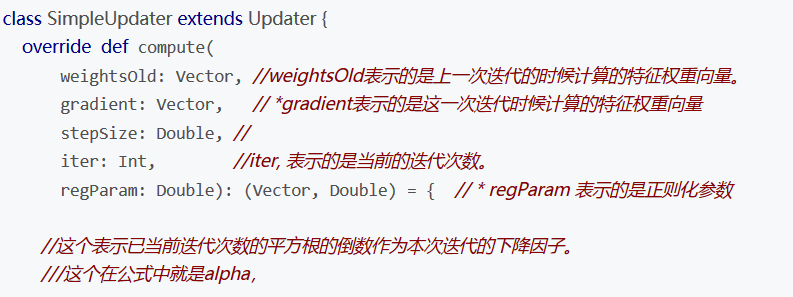


图 4-16 代码

* **第八步：最后就是线性回归模型类，LinearRegressionModel类。weigths表示的是权重值，intercept表示的是偏置项，常数项。有一个predictPoint方法用来预测y值。部分代码如下：**

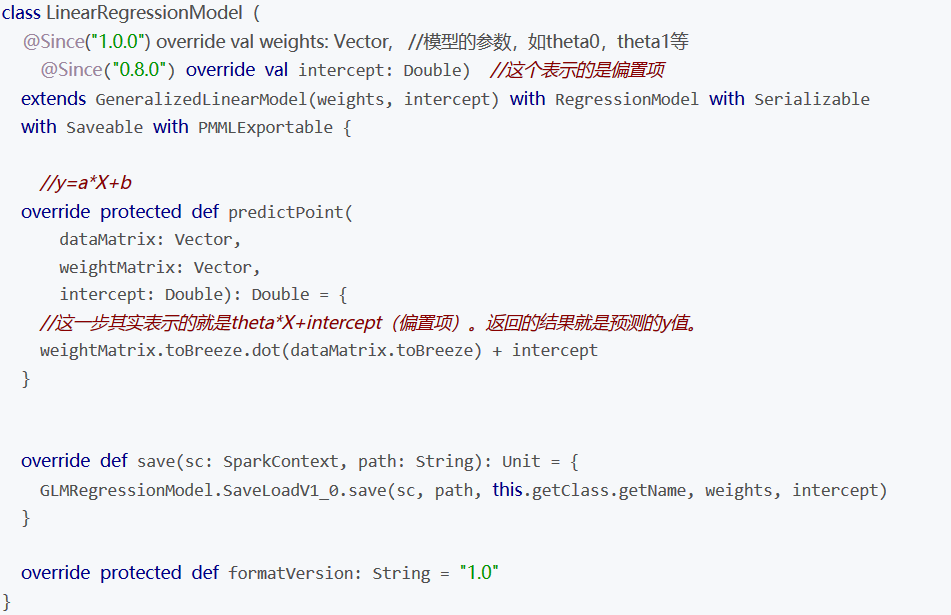


图 4-17 代码

## 4.7第三阶段：数据分析④(基于内容过滤和协同过滤的推荐算法)

### 4.7.1 混合推荐算法技术特色

1）对传统的基于内容的推荐算法进行了改进，提出了基于内容过滤的推荐算法。该算法的基本思想是：充分利用求职者注册的简历信息和招聘企业发布的招聘信息，将二者进行比较，计算它们之间的相似度。该算法解决了在系统用户没有操作行为时的冷启动问题。

2）为充分利用系统用户的注册信息和操作行为信息进行推荐，引入了协同过滤算法，同时根据网络招聘领域推荐系统的特点将传统的协同过滤算法进行了改进，将基于双重聚类的协同过滤推荐算法运用到了该推荐系统中，对数据稀疏性问题有了一定的改善。

3）将基于内容过滤的推荐算法和基于双重聚类的协同过滤推荐算法进行融合，用融合后的混合推荐技术作为系统的推荐引擎。这样就同时解决了传统推荐算法面临的冷启动、数据稀疏性等问题。

### 4.7.2 基于内容过滤的推荐模型

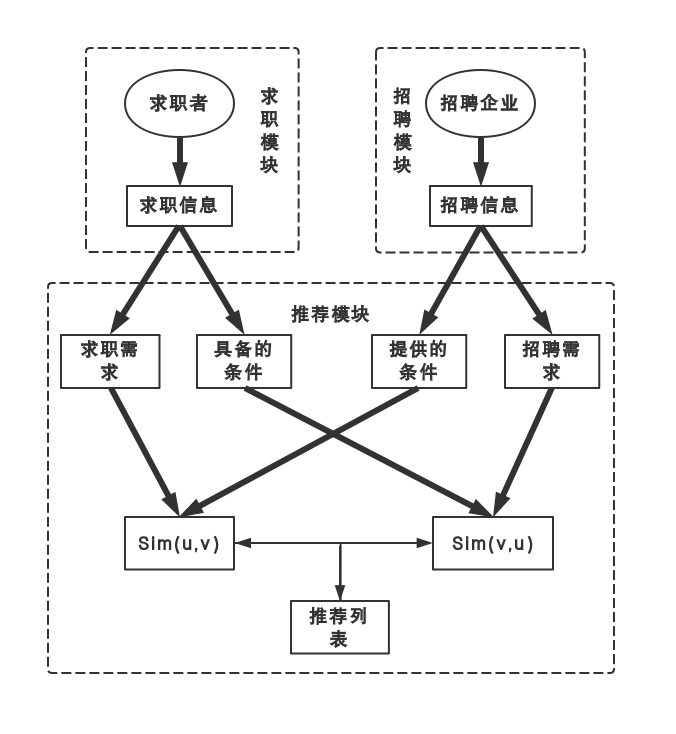


图 4-18 招聘模型

1）求职模块：该模块负责向求职者进行信息采集（包括毕业院校、专业、学历、专业技能、外语技能、工作经验、年龄、性别、是否适应出差等。），求职者可以通过此模块来输入自己的求职信息。

2）招聘模块：该模块主要负责管理已经收集到的招聘企业发布的职位招聘信息（包括工作地点、工资、福利、工作环境、职位发展、工作内容等）。

3）推荐模块：该模块负责对双方的信息进行分析和处理，按着一定的匹配规则，对数据库中的属性描述和偏好记录进行相似度匹配计算，然后根据返回的匹配结果，向求职者进行职位推荐。

### 4.7.3 文本分词

通过对求职者或招聘者提供的信息解析和分类，可以看出如求职意愿和招聘意愿等属性为中文文本信息，可能蕴含重要的求职或招聘意愿信息。因此，需要对这些中文文本信息进行进一步的中文分词处理。中文分词是对中文进行自然语言处理的重要步骤。英文用空格作为单词之间的分隔符，而中文是连在一起的，只能通过句和段作为符号的分割，没有形式上的分界符。

我们采用了基于字符串匹配的分词方法，这种分词方法是将待分词的汉字字符串与一个机器词典相对比，如果能在该词典中找到一个字符串，则认为成功识别出该词。

### 4.7.4 文本相识度计算

为了对职位信息和招聘信息进行更好的解析，从而建立用户偏好模型，在对文本属性进行分词后，还应进行文本相似度的计算。向量空间模型（VSM）是一种比较常用的处理自然语言的模型，该模型将具有文本属性的信息转化为计算机可以计算的结构化数据，并通过向量之间的相似度来反映文本信息之间的相似度如图 4-19。

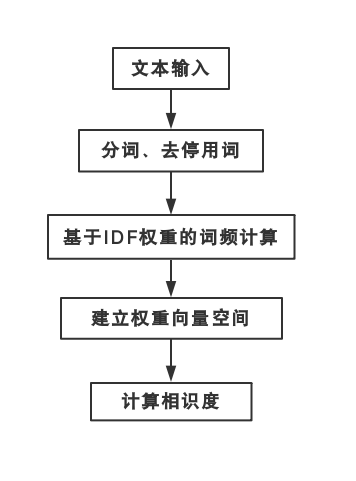


图 4-19 文本相识度计算过程

图 4-19 表示了用向量空间模型来计算文本相似度的过程，首先对语料进行中文分词、去除停用词后生成词条串；然后通过计算词条串中不同的词条出现的频率来对各个词条的权重进行赋值，从而建立向量空间模型，最后计算特征向量的余弦值来表示文本之间的相似度。

具体步骤：

* **预处理**

由于在文本语料中会存在一些对文本内容识别意义不大但出现频率很高的词、标点、符号，在进行相似度计算时需要先对语料进行预处理。对语料的预处理过程包括中文分词和去掉停用词，具体预处理过程如下图。

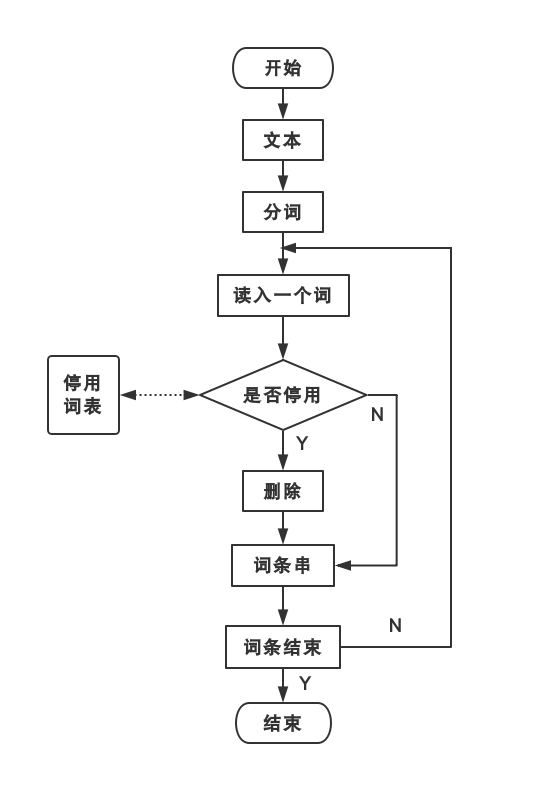


图 4-20 语料的预处理过程

图 4-20 表示了对语料的预处理过程，在对文本信息进行中文分词后，每读入一个词时，判断是否为停用词，将去掉停用词后的词条串保存下来，直到词条结束。

* **权重值计算**

通过以上的预处理过程，将文本转换为一组词条后，需要根据每个词条的重要程度的不同来分别设置不同的权重值。

* **相似度计算**

在向量空间模型中，通过余弦值来表示两个文本d1和d2之间在内容相关度向量空间模型能用简单向量的形式将非结构化的文本内容转化成易于进行数据运算的结构化形式，并且能够体现出概念在用户偏好中不同的重要程度，该模型在对自然语言的处理领域得到了广泛的应用。

### 4.7.5 匹配度计算

基于内容过滤的推荐算法中是通过计算选择求职者和企业双方的偏好模型的匹配度来向用户展示推荐列表。为了使匹配的成功率尽可能的提高，来实现个性化推荐系统中的推荐，不仅需要获取用户对项目的偏好，还应该把项目看成具有选择意思的主体，来捕获其的偏好信息。在推荐计算中，对于两个对象，当二者能够匹配时，则计算两者的相似度匹配的概率，然后按概率的大小依次选取最合适的匹配项目进行推荐，即通过寻找最优匹配来为用户进行推荐。现在研究求职者与招聘岗位之间的选择问题。在某一个时间段内，当系统中存在 m 个求职者和 n 个招聘岗位，如何找到一种匹配方案使这些用户之间能够实现最大匹配，即推荐成功率最高。这里的成功率，是指求职者成功选择招聘岗位的概率，也表示了求职者和企业岗位的偏好程度，双者的偏好程度越高，求职者找到心仪工资成功的机会也就越大。

### 4.7.6 基于双重聚类的协同过滤推荐

由于基于内容的推荐算法没能充分利用用户的行为信息，且属于静态推荐方法，推荐结果不能随着用户兴趣的漂移而改变，不能满足个性化推荐要求。对此，我们还引入了基于双重聚类的协同过滤推荐算法，两种推荐方法共同使用，以满足网络求职领域的个性化推荐要求。

协同过滤推荐技术的产生来源于实际生活中的这样一种现象:若一人购买了某件商品后，对这件商品特别满意，该用户朋友圈的人也很可能会购买同样的商品或跟其类似的商品。因此，使用协同过滤推荐的关键任务是寻找与目标对象相似的邻居对象，然后对该用户推荐其邻居所喜爱的项目。这种推荐方式不用关心项目的具体内容，因此应用于项目内容不易解析的领域，同时容易发现用户潜在偏好，从而实现跨类别的推荐，为用户推荐新颖的项目，能给用户带来惊喜，从而得到用户的认可。

### 4.7.7 双重聚类的协同过滤推荐

在推荐系统的研究中，数据稀疏性问题始终是面临的一个重要问题，特别是在协同过滤推荐系统中，数据的稀疏通常会对算法的性能产生严重影响。在协同过滤中需要计算用户之间的相似度，而相似度的计算是基于用户-项目评分矩阵的。在一个推荐系统中往往用户和项目的数据都是巨大的，但是一个用户的精力是有限的，都只能对一小部分项目产生行为，这必然导致用户-项目评分矩阵的严重稀疏。用户-项目评分矩阵的稀疏情况如表xxxx所示。

表 4-3 用户-项目评分矩阵稀疏情况

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 项目1 | 项目2 | 项目3 | 项目4 | 项目5 |
| 用户1 | 3 |  |  | 5 |  |
| 用户2 |  |  | 4 |  | 1 |
| 用户3 |  | 1 |  |  |  |
| 用户4 | 2 |  |  |  |  |
| 用户5 |  |  |  | 4 |  |

在这样稀疏的评分矩阵中，在计算用户间相似度时由于用户之间的共有评分数据极少，计算得到的相似度精度也就非常差。要解决数据稀疏性对相似度精度的影响，需要对用户-项目评分矩阵进行填充。填充后若可以达到两两用户间的共有评分项增加，就可以使相似度精度提高。如表xxxx展示了用户‐项目评分矩阵稠密的情况。

表 4-4 用户-项目评分矩阵数据填充后情况

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 项目1 | 项目2 | 项目3 | 项目4 | 项目5 |
| 用户1 | 3 | 3 |  | 5 | 3 |
| 用户2 |  | 4 | 4 |  | 1 |
| 用户3 |  | 1 |  | 1 |  |
| 用户4 | 2 |  | 2 |  | 2 |
| 用户5 |  | 4 |  | 4 |  |

常用的评分预测的方法有基于邻域的方法，基于聚类的方法和数据降维的方法。我们针对网络招聘领域内用户内容可解析的特征引入了基于用户和项目双重聚类的方法，分别对求职者和招聘岗位的信息解析后进行聚类，以解决数据稀疏性的问题。该推荐算法流程图如图 4-1 所示。

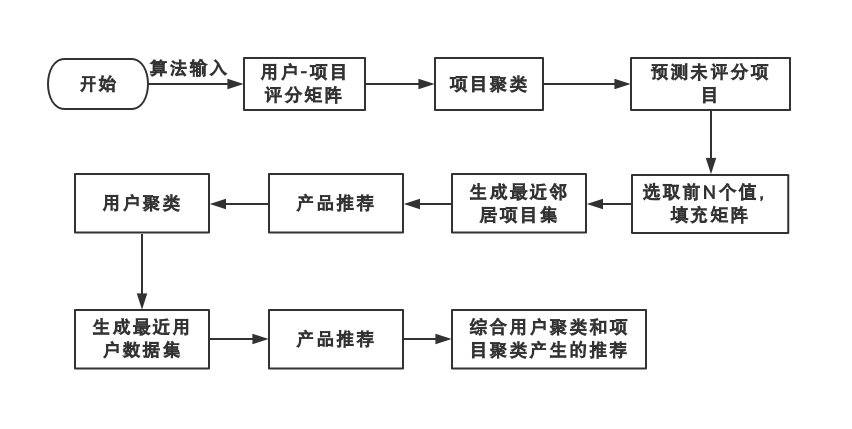


图 4-20 双重聚类协同过滤算法流程图

* **聚类分析**

我们采用了比较经典的聚类算法——K-means 算法，分别对求职者和招聘职位进行聚类计算。K-means 算法作为聚类算法的一种，具有方便且高效的特点，是目前在数据挖掘中广泛使用的一种算法。K-means 算法首先从数据中随机选择 K 个对象作业初始聚类的中心，然后计算数据中所有数据分别到每个聚类中心的距离，如将对象聚为 K 类，则全体数据需要计算 K 次，之后分别将与每个聚类中心距离最近的对象分为一类，分别计算每个类中的对象的中心值，并作为形成新的聚类的中心值。通过不断计算出新的聚类集合和聚类中心，若每个类中的聚类中心值不变，或变化在规定的阈值以内，则停止聚类计算。

在对求职毕业生和招聘岗位的聚类计算中，由于可以通过对毕业生求职信息和招聘岗位招聘信息的解析可获取他们的选择意愿，从而建立用户的偏好向量模型，在K-means算法的距离计算过程中，可采用对用户偏好向量的夹角余弦值来表示对象同聚类中心的距离。



* **预测未评分项目的评分**

为了使用户-项目评分矩阵变得稠密，需要预测用户对未评分项目的评分值，并将预测的评分值填入用户-项目评分矩阵中。然后对聚类后的每一个用户-项目矩阵，使用基于项目的协同过滤算法。算法过程如下：

1）在聚类后的用户-项目矩阵Rtxv中，对于每一个项目i，计算该项目与其它项的相似度，选取相似度最高的 NI 个项作为该项目的邻居集。相似度计算公式如下：



其中，和分别表示在聚类后的评分矩阵中所有用户对项目 a 和项目i的评分平均值；为项目a和项目i在项目集中都有过评价的用户子集；表示在用户-项目矩阵中用户 i 对项 j 的评分值。

1. 根据目标项目已有的评分值和邻居项的评价信息，未评价项的评分值进行评分预测，并取预测值中最大的 N 个项目作为推荐评价集合。预测值计算公式如下：



3）其中，表示项目 a 与项目 v 的相似度；表示用户 v 对项目 k 的已有的评分值。

4）在用户-项目矩阵中填入预测评分值集合，从而形成新的用户-项目矩阵。预测评分值填入后，评分矩阵值的数据稀疏程度将大幅度得到改善，使得推荐效果得到提高。

* **双重聚类并进行推荐**

1. 在分别进行了项目的聚类和对评分矩阵的填充后，对数据稀疏程度得到改善的用户-项目评分矩阵使用传统的协同过滤算法进行推荐。对每一个用户i，选取他感兴趣的I个矩阵，即包含较多这个用户的评价项的矩阵，作为对用户i 的推荐产生数据源矩阵，然后使用基于用户的协同过滤算法，为该用户计算推荐项。具体过程如下：

* 首先，计算在用户-项目矩阵中用户i与其他用户的相似度，将相似度最高的 NU 个用户作为该用户的邻居。然后，根据用户i已有的评分值和其 NU 个最近邻居的评价信息，计算该用户对其未评分项目的评分预测值，将预测值中最大的 N 个作为推荐评价集合。
* 然后，合并推荐评价集合，并从合并后的推荐评价集合中选取前 N 个预测值组成最终推荐评价集合。
* 最后，提取推荐评分集合里的项目，并生成用户 i 的推荐项目集合。

2）将用户聚类后进行推荐的过程与之前项目聚类进行推荐过程类似。对于目标用户i,选取该用户所在的I个矩阵作为用户-项目评分矩阵输入，之后使用协同过滤算法对该用户进行推荐。具体算法与上一小节相似。

3）在对于目标用户 i 进行推荐计算时，结合项目聚类产生的推荐和用户聚类产生的推荐结果，当两推荐结果中有相同项目，则进行加权求和，最后选取 N 个评分预测值取大的 N 个项目组成最终的推荐项目集合。

### 4.7.8 个性推荐总体设计

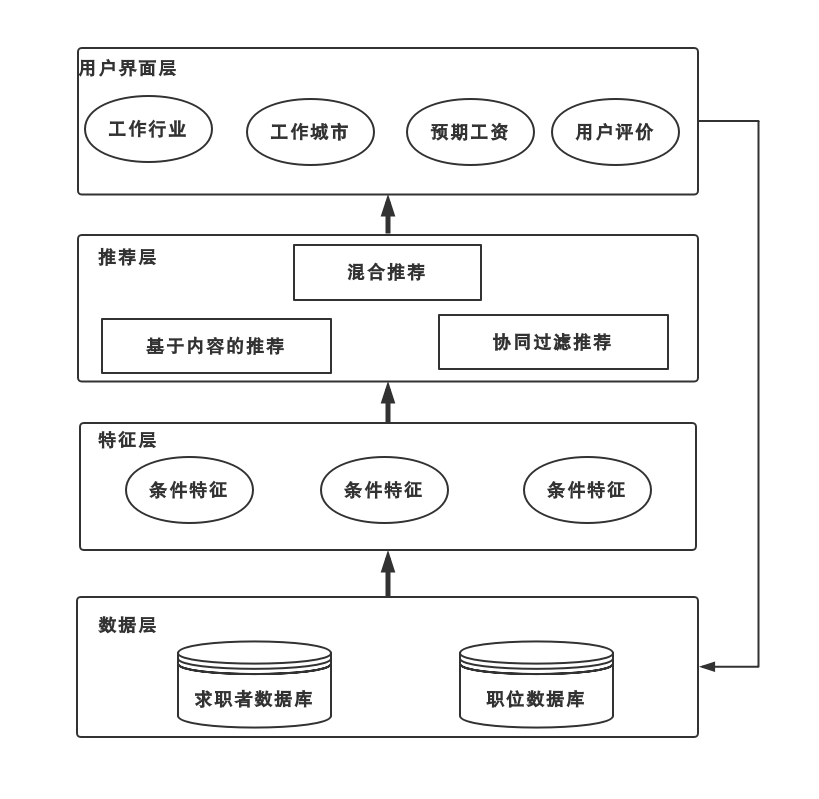


图 4-21 个性推荐模型

* **数据层**

在推荐系统中，数据层从数据库中获取求职者的简历信息、注册信息、职位的招聘信息和招聘者的注册信息，同时从 WEB 服务器的日志文件中获取用户的浏览、职位申请等行为信息，将这些信息进行预处理后，建立面向个性化推荐功能的数据仓库。数据层对推荐系统的原始信息进行维护和管理，保证推荐算法的输入信息准确可靠。

* **特征层**

在采集到用户的求职与招聘信息及行为信息后，需为用户建立偏好特征。在本系统中分别建立毕业生及招聘岗位的要求特征、具备条件特征及行为偏好特征。用户兴趣特征的建立需要对用户信息进行解析后，选用相关的属性值，并分别设置不同的权重。

* **推荐层**

本文所设计的推荐系统采用了基于内容的互惠推荐与协同过滤推荐相结合的混合方式进行推荐。分别对两种推荐结果进行加权后输出推荐列表。考虑到两种推荐算法的优缺点，即在用户登录系统初期，其行为数据较少，而协同过滤推荐算法受到数据稀疏性的影响较大，而基于内容的互惠推荐不受该影响，因此，此时基于内容的推荐算法应具有较高的权重。而随着用户的行为数据增多、使用系统的时间较长后，用户的兴趣可能会发现漂移，此时，协同过滤的推荐算法具有较高的权重。

* **用户界面层**

用户界面层是用户与系统进行交互的窗口，主要有用户的注册、填写信息、浏览、申请等交互行为。同时，将计算出的推荐列表展示给用户。

## 4.8第四阶段：数据导出

### 4.8.1 数据导出流程

* **列出mysql数据库中的所有数据库命令**  
   sqoop list-databases --connect jdbc:mysql://localhost:3306/ --username root --password 123456
* **连接mysql并列出数据库中的表命令**  
   sqoop list-tables --connect jdbc:mysql://localhost:3306/FuChuang --username root --password 123456
* **将关系型数据的表结构复制到hive中**  
   sqoop create-hive-table --connect jdbc:mysql://localhost:3306/FuChuang --table main\_data --username root --password 123456 --hive-table main\_data
* **从关系数据库导入文件到hive中**sqoop import --connect jdbc:mysql://localhost:3306/FuChuang --username root --password mysql-password --table t1 --hive-import
* **将hive中的表数据导入到mysql中**Sqoop export --connect jdbc:mysql://localhost:3306/FuChuang --username root --password admin --table main\_data --export-dir/user/hive/warehouse/uv/dt=2019-02-26

## 4.9第五阶段：数据可视化

### 4.9.1 通过Bootstrap进行数据可视化的优势

为了让我们的展示界面更加直观、清晰、简洁，我们采用了前端框架bootstrap，该框架基于层叠样式表（css），java脚本，以及html。并拥有许多优势：

* 扩展性强，能够更好的与现实的web开发项目结合。
* 比较成熟，在大量的项目中充分的使用和测试。
* 拥有现成UI组件，可以快速搭建网页页面。
* 前端处理简洁，在Blog系统搭建过程中使用方便。
* Bootstrap框架为用户提供了一套响应式移动设备优先的流式栅格系统，拥有完备的框架结构，整体效果和谐，对谷歌，火狐，IE等浏览器均可支持，项目开发方便快捷。

### 4.9.2 ECharts图表展示

在数据展示层面，为了给用户展示更直观，更有逻辑的数据，我们应用了ECharts来进行数据的整合和展示。

ECharts，一个纯 Javascript 的图表库，可以流畅的运行在 PC 和移动设备上，兼容当前绝大部分浏览器（IE8/9/10/11，Chrome，Firefox，Safari等），底层依赖轻量级的 Canvas 类库 ZRender，提供直观，生动，可交互，可高度个性化定制的数据可视化图表。同时具有很多优势：

* 丰富的图表类型

ECharts为我们提供了常规的折线图（例如：热门岗位人数与工资关系图）、柱状图（北京实时人员学历招聘需求图）、饼图（例如：应届生就业方向占比图）等等。并且支持图与图之间的混搭。

* 动态数据

ECharts由数据驱动、数据的改变驱动图表展现的改变。因此动态数据的实现也变得异常简单。

* 大数据量的展示

我们展示的数据量非常大，如何才能使这些数据发挥它应有的价值。ECharts对大数据的处理能力非常好，借助Canvas的能力，ECharts能够轻松的展示这些数据。

* 性能较高

虽然ECharts中的canvas计数只支持2D绘图，不支持3D绘图。但是canvas绘图不会给每个点生成对象，因此绘制图形可以非常快，占用内存特别少，兼容性也比较好。

### 4.9.3 ECharts图表展示的流程

1）引入Echarts



图 4-22 代码

2）创建一个装ECharts的DOM容器



图 4-23 代码

3）创建热门行业招聘需求的柱状图

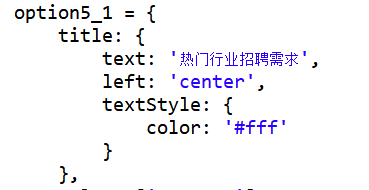


图 4-24 代码

4）后台返回数据

### 4.9.4 Ajax数据交互

Ajax 异步JavaScript+XML，依赖的是现有的CSS/HTML/Javascript，而其中最核心的依赖是浏览器提供的XMLHttpRequest对象。这个对象为向服务器发送请求和解析服务器响应提供了流畅的接口，使得浏览器可以发出HTTP请求与接收HTTP响应，实现在页面不刷新的情况下和服务端进行数据交互

### 4.9.5 Ajax进行数据交互的流程

1）填写地址（url）

要求为String类型的参数，（默认为当前页地址）发送请求的地址  
2）返回结果

Success：要求为Function类型的参数，请求成功后调用的回调函数，有两个参数。

Error：要求为Function类型的参数，请求失败时被调用的函数。  
3）解析结果  
4）展示数据

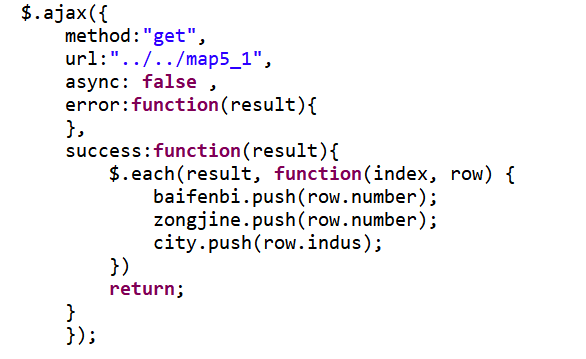


图 4-25 代码

## 4.10第六阶段：系统管理①（数据后台管理模块）

页面采用bootstrap框架，结合后端的SSM框架实现了整个的数据后台管理模块。实现原理是调用JavaAPI操控集群实现数据处理同时把HDFS和yarn管理界面内嵌到网页中。

### 4.10.1 bootstrap框架介绍

为了让我们的展示界面更加直观、清晰、简洁，我们采用了前端框架bootstrap，该框架基于层叠样式表（css），java脚本，以及html。并拥有许多优势：

1)扩展性强，能够更好的与现实的web开发项目结合。

2)比较成熟，在大量的项目中充分的使用和测试。

3)拥有现成UI组件，可以快速搭建网页页面。

4)前端处理简洁，在Blog系统搭建过程中使用方便。

5)Bootstrap框架为用户提供了一套响应式移动设备优先的流式栅格系统，拥有完备的框架结构，整体效果和谐，对谷歌，火狐，IE等浏览器均可支持，项目开发方便快捷。

### 4.10.2 SSM框架介绍

SSM框架，是Spring + Spring MVC + MyBatis的缩写，Spring依赖注入DI来管理各层的组件，使用面向切面编程AOP管理事物、日志、权限等。SpringMVC代表了Model(模型)View(视图)Controller(控制)接收外部请求,进行分发和处理。Mybatis是基于jdbc的框架,主要用来操作数据库,并且将业务实体和数据表联系起来。

### 4.10.3 SSM 框架流程介绍

1)客户端发送请求到DispacherServlet（分发器）

2)由DispacherServlet控制器查询HanderMapping，找到处理请求的Controller

3)Controller调用业务逻辑处理后，返回ModelAndView

4)DispacherSerclet查询视图解析器，找到ModelAndView指定的视图

5)视图负责将结果显示到客户端

## 4.11第六阶段：系统管理②（基于Android的移动端控制模块）

### 4.11.1 Android介绍

Android是基于Linux系统内核开发设计的，Google在内核之上开发了自己的Dalvik Java虚拟机。由于采用Java虚拟机，所以在Android上开发原生App用的开发语言是Java。目前为止Android已经是全球最大的智能手机操作系统，最新版本是Android8.0。Android所具有的优势或者特性：

* 开源
* 多元化设备支持
* Dalvik 虚拟机
* 开放的第三方应用
* 无缝和google集成

### 4.11.2 Butterknife模版代码生成工具

* **Butterknife简介**

Butter Knife是一款基于注解来生成模板代码的安卓工具。通过它我们可以更简便的绑定页面布局和添加点击事件的监听。使用@BindView 方法来代替findViewById。此外还可以通过@BindBool,@BindColor,@BindDimen,@BindDrawable,@BindInt,@BindString来绑定对应的资源。在集合或数组中存放多个view,在行为、设置、属性上设置一次，所有的view都能生效。使用@OnClick方法来添加点击事件的监听由于Butter Knife的实现方式是生成代码而非反射的方式，所以并不会减慢app的运行速度。而使用它可以在一定程度上简化代码。

* **Butterknife技术流程**

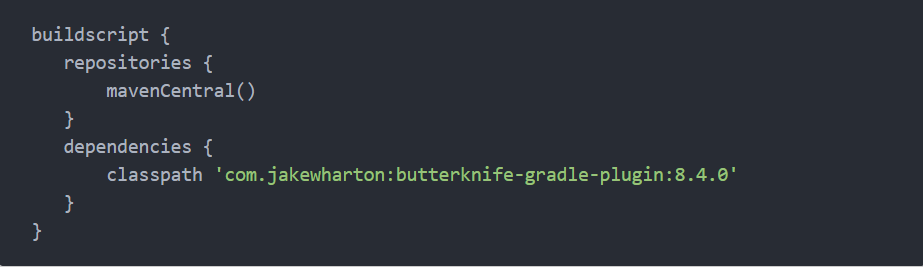


图 4-26 在projec的.gradle中添加路径

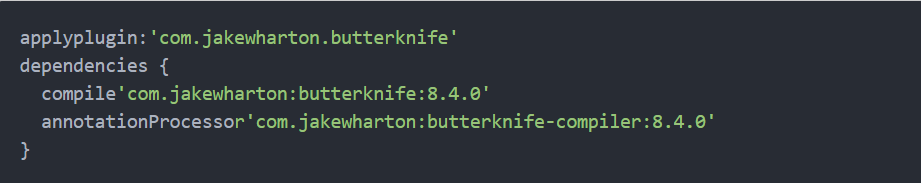


图 4-27 在module的gradle中添加依赖

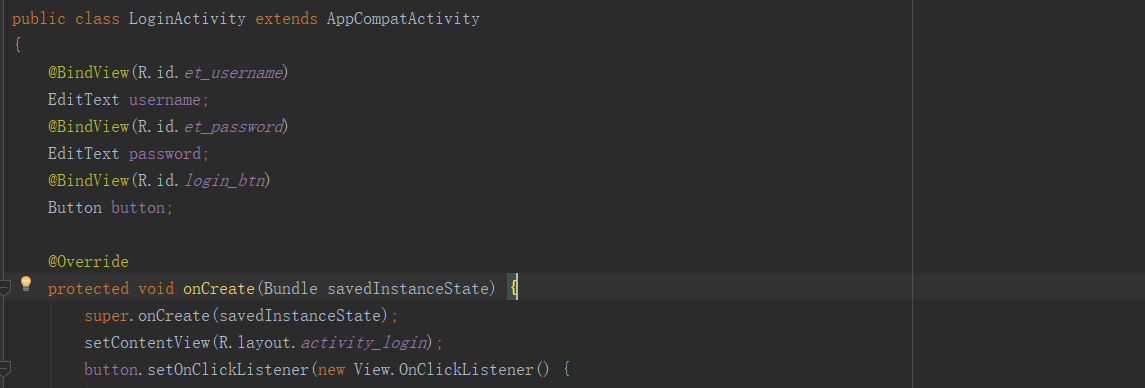


图 4-28 绑定view

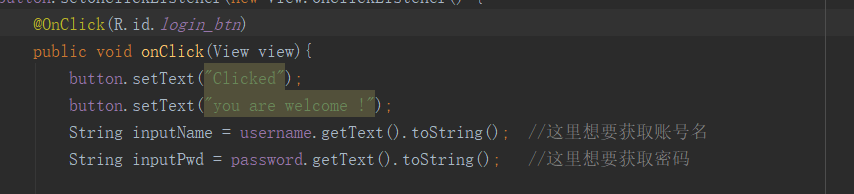


图 4-29 绑定监听器

### 4.11.3 Rxjava响应式编程

* **Rxjava简介**

Rx（ReactiveX，响应式编程）是一种事件驱动的基于异步数据流的编程模式，整个数据流就像一条河流，它可以被观测（监听），过滤，操控或者与其他数据流合并为一条新的数据流。而RxJava是.Net Rx在JVM上的实现。RxJava最核心的两个东西是Observables（被观察者，事件源）和Subscribers（观察者），Observables发出一系列事件，Subscribers处理这些事件。

* **Rxjava技术流程**

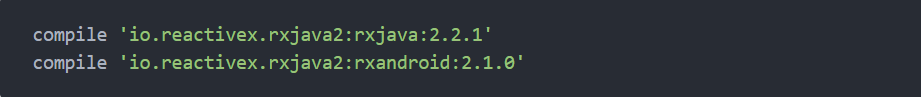


图 4-30 在module 的gradle中添加依赖

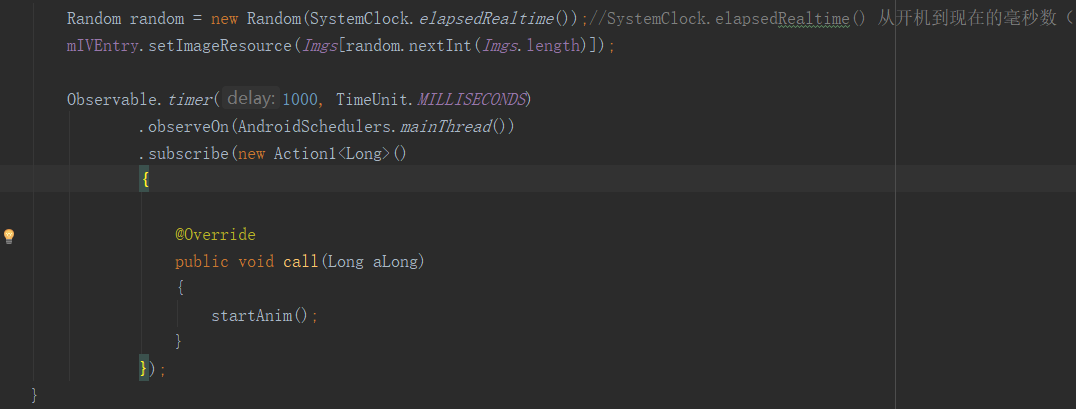


图 4-31 利用observable设置图片加载的一段延迟（延迟为1秒）

### 4.11.4 Glide图片加载库

* **Glide简介**

Glide是一个快速高效的Android图片加载库，注重于平滑的滚动。Glide提供了易用的API，高性能、可扩展的图片解码管道（decode pipeline），以及自动的资源池技术。

* **Glide技术流程**

Glide 支持拉取，解码和展示视频快照，图片，和GIF动画。Glide的Api是如此的灵活，开发者甚至可以插入和替换成自己喜爱的任何网络栈。默认情况下，Glide使用的是一个定制化的基于HttpUrlConnection的栈，但同时也提供了与Google Volley和Square OkHttp快速集成的工具库。虽然Glide 的主要目标是让任何形式的图片列表的滚动尽可能地变得更快、更平滑，但实际上，Glide几乎能满足你对远程图片的拉取/缩放/显示的一切需求。

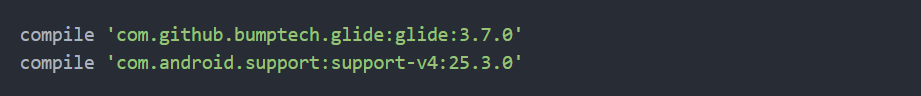


图 4-32 在Module的Gradle添加依赖

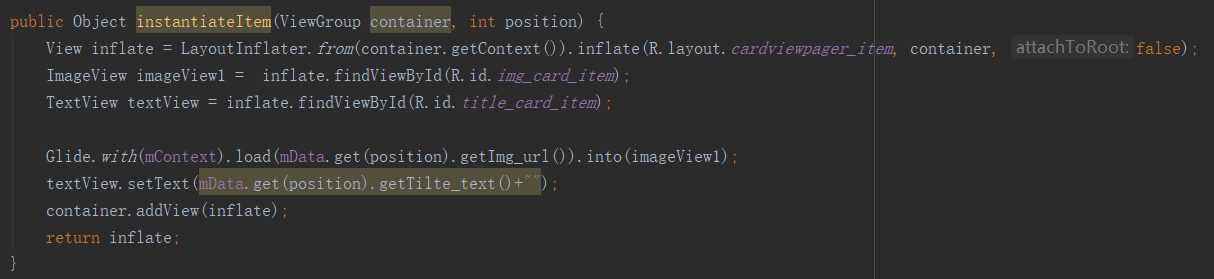


图 4-33 处理图片在原图的基础上变为圆角

### 4.11.5 **Android Library**

* **Android Library简介**

Android Library在目录结构上与Android App相同，它能包含构建APP所需的一切（如源代码、资源文件、Android Manifest）。Android App最终被编译打包成能在Android设备上运行的APK文件，但是Android Library被编译成供其它Android App依赖的Android Archive (AAR)文件。

* **Android Library技术流程**

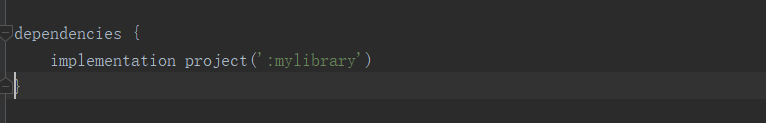


图 4-34 Library创建成功后在project的gradle中添加项目依赖

在本项目的library中，主要负责的任务是为在两张图片切换时，提供一个固定缩放比例的效果。且使得荧幕中的两边两个 item 被缩小且透明度变低。代码如图 所示：

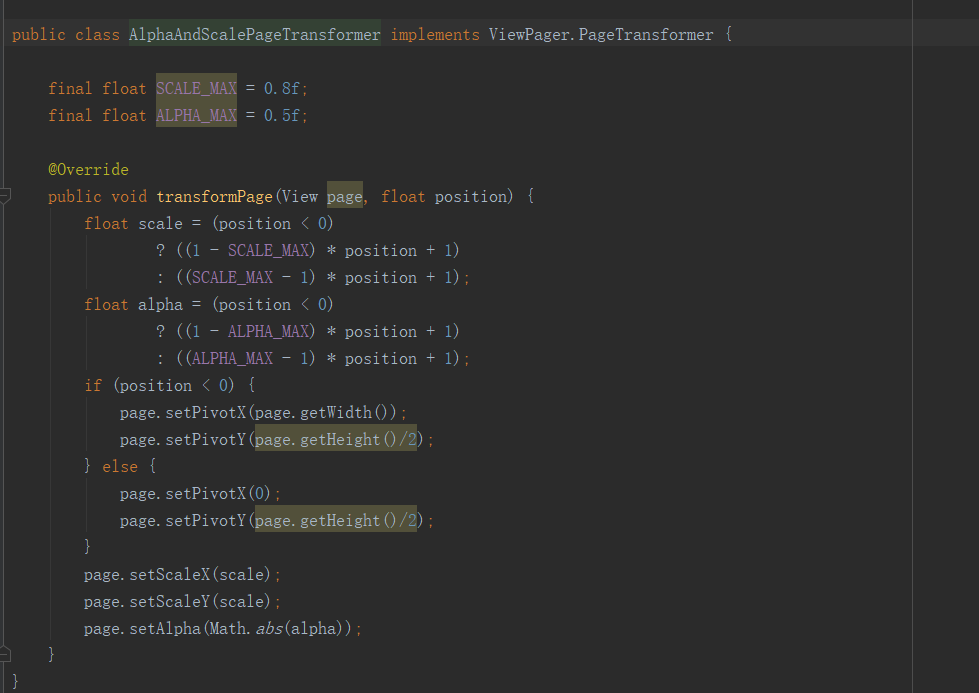


图 4-35 代码

### 4.11.6 网络编程技术

* **网络编程简介**

Android应用作为一个客户端程序，其绝大部分都是需要进行网络请求和访问的，而http通信是一种比较常见并常用的通信方式。在Android中http网络编程中有两种实现方式，一种是使用HttpURLConnection，另一种就是使用HttpClient。这两种实现方式的大体过程都是：Android客户端向服务器发出请求，服务端接收请求并响应，服务端返回数据给客户端。在Http通信中有POST和GET两种方式，其不同之处在于GET方式可以获得静态页面，同时可以将请求参数放在URL字符串后面，传递给服务器；而POST方式的参数则是放在Http请求中的。因此，无论你是用哪一种HTTP实现方式都需要明确所使用的请求方式。

* **网络编程技术流程**



图 在AndroidManifest.xml添加网络访问权限

HttpClient是Apache Jakarta Common下的子项目，用来提供高效的、最新的、功能丰富的支持HTTP协议的客户端编程工具包，并且它支持HTTP协议最新的版本和建议。在获得HttpURLConnection对象之后，采用GET方式请求数据，此时即可使用输入流读取信息和发送信息。当返回编码为200的时候证明数据访问成功。代码如图 所示：

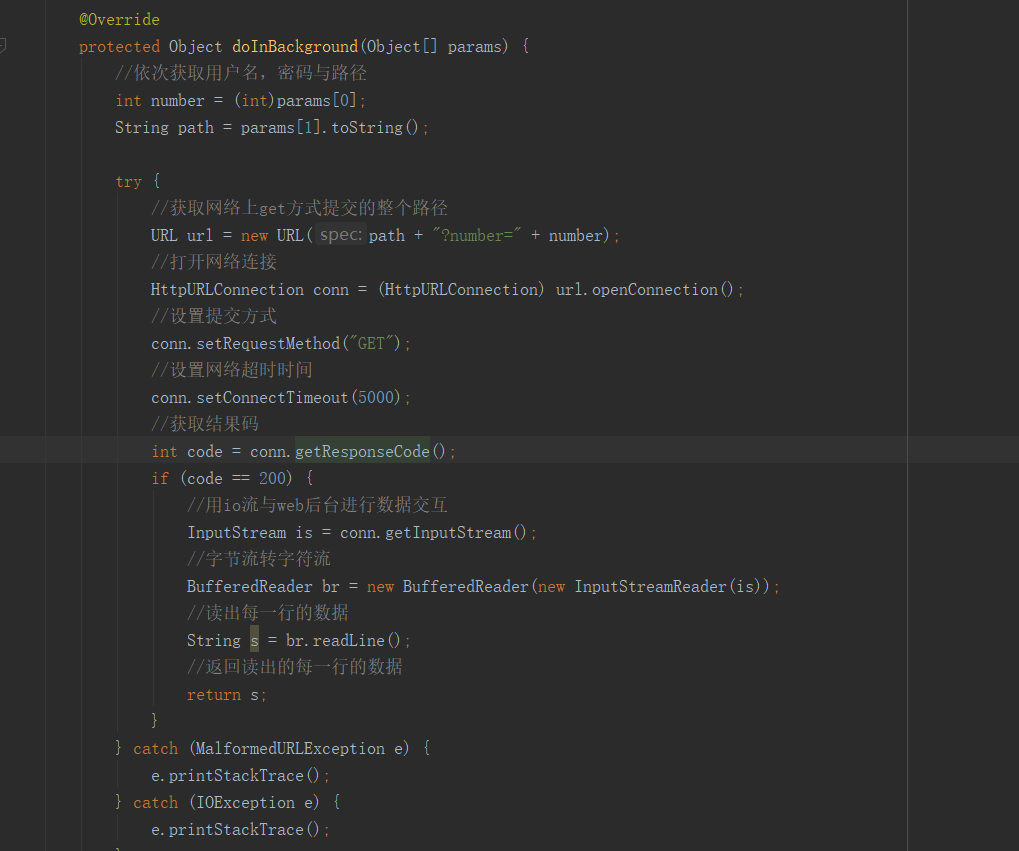


图 4-36 代码

# 详细设计说明

## 5.1 用例图

* 管理端的功能包括数据获取，数据加载更新，集群数据监控，用户管理，以及集群节点监控和HDFS文件系统管理。数据获取主要包括数据爬取，数据清洗，数据保存。数据加载更新主要功能是将爬取的数据添加到集群中。集群数据监控主要包括监控历史数据量，监控实时数据量，监控数据来源。集群节点监控主要功能是监控节点状态，当集群中某一节点出故障时，能够及时发现并采取行动。用户管理主要包括添加用户，删除用户，查看用户信息。HDFS文件系统管理只要包括下载数据TXT文件和HDFS文件管理的功能。

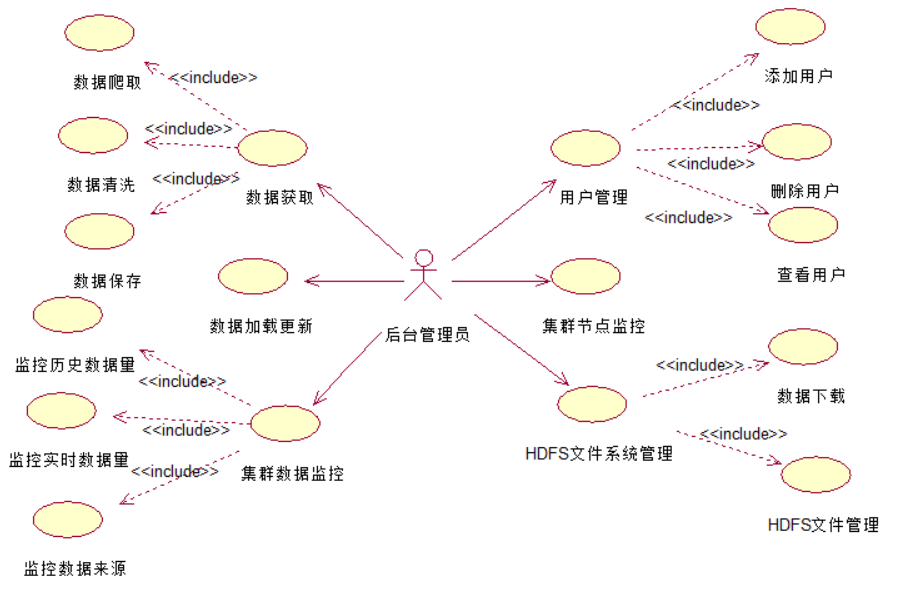


图 5-1 用例图一

* 管理端的功能包括数据获取，数据加载更新，集群数据监控，用户管理，以及集群节点监控和HDFS文件系统管理。数据获取主要包括数据爬取，数据清洗，数据保存。数据加载更新主要功能是将爬取的数据添加到集群中。集群数据监控主要包括监控历史数据量，监控实时数据量，监控数据来源。集群节点监控主要功能是监控节点状态，当集群中某一节点出故障时，能够及时发现并采取行动。用户管理主要包括添加用户，删除用户，查看用户信息。HDFS文件系统管理只要包括下载数据TXT文件和HDFS文件管理的功能。

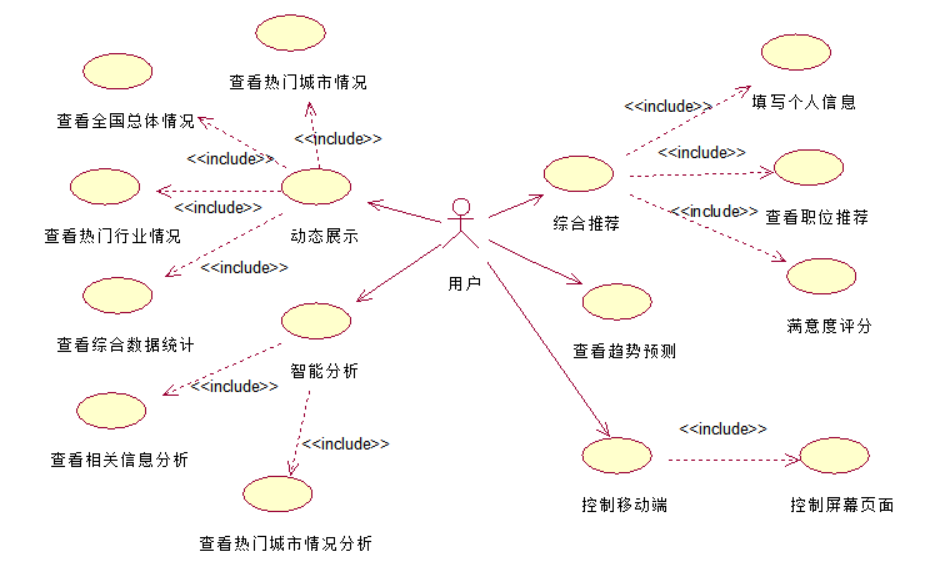


图 5-2 用例图二

## 5.2 顺序图

* 对招聘信息可视化分析业务具体实现是管理员先运行爬虫程序，通过关键字将招聘信息爬取下来，然后进行数据清洗，将可用数据筛选出来，并保存到数据库。管理员将数据库的招聘信息导入到集群中，先进行数据预处理，将脏数据和重复数据去除，再分别通过HBASE数据仓库进行实时数据分析和Hive数据仓库进行离线数据分析，然后将分析结果分别保存到HDFS文件系统，之后通过sqoop数据迁移工具将结果导入MySQL数据库。管理员将过去几个月获得的招聘数据导入到集群，运行Spark MLlib预测算法得到2019年的招聘趋势，并将结果保存到MySQL数据库中。最后将MySQL数据库中的相关结果数据转存为JSON格式，用Ajax异步获取数据，再通过ECharts将结果展示出来。

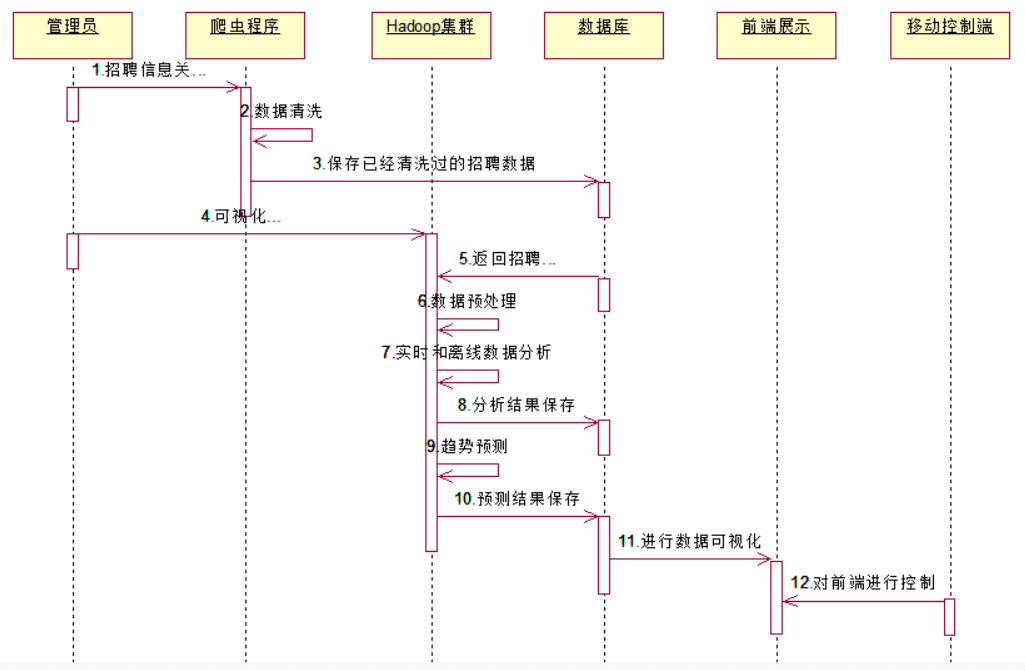


图 5-3 顺序图一

* 集群在搭建成功以后，需要对集群的数据获取，数据加载更新，集群数据监控，集群节点，用户信息进行管理。由于功能繁杂，我们开发了后台集群管理系统，可以大大简化管理员的工作，相关管理过程如上图所示。

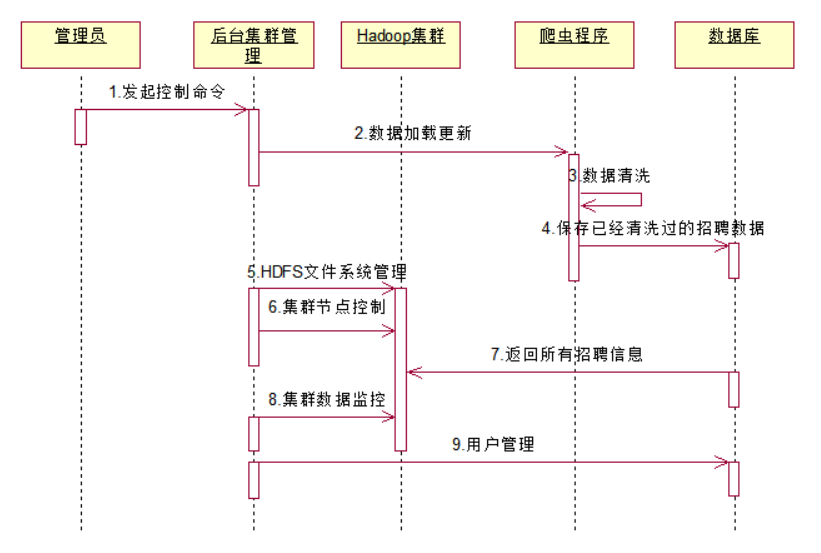


图 5-4 顺序图二

* 综合推荐是我们最为重要的一个功能点，它可以充分利用求职者填写的信息和招聘企业发布的招聘信息，将二者进行比较，计算它们之间的相似度，将相识度高的招聘信息推荐给求职者。相关推荐过程如上图所示。

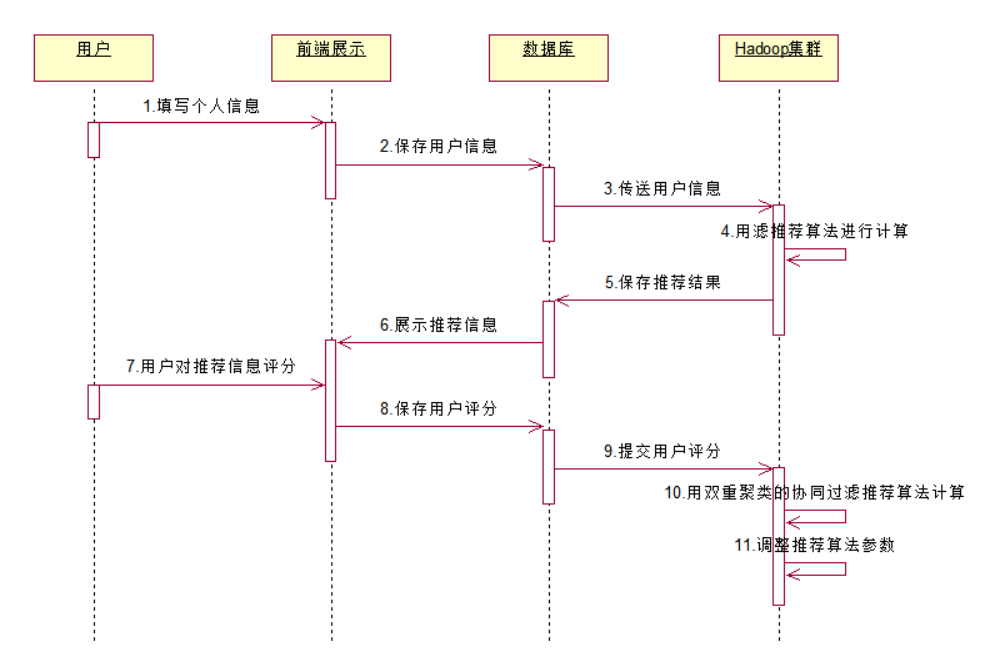


图 5-5 顺序图三

## 5.3 数据输入信息表

表 5-1 全国总体情况页面数据输入信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 长度 | 所属图表 |
| 城市 | name | varchar | 255 | 行业招聘  热度地图 |
| 就职人数 | value | int | 255 |
| 城市 | name | varchar | 255 | 全城市  招聘需求图 |
| 招聘人数 | value | int | 255 |
| 行业 | name | varchar | 255 | 全行业  工资状况图 |
| 招聘人数 | value | int | 255 |
| 最高工资 | maxs | double | 255 |
| 最低工资 | avg | double | 255 |

表 5-2 热门城市情况页面数据输入信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 长度 | 所属图表 |
| 行业 | name | varchar | 255 | 北京实时  行业需求 |
| 招聘人数 | value | int | 255 |
| 行业 | indus | varchar | 255 | 北京实时热门  行业工资状况图 |
| 最高工资 | max | double | 255 |
| 平均工资 | avg | double | 255 |
| 最低工资 | min | double | 255 |
| 学历 | name | varchar | 255 | 北京实时人员  学历招聘需求 |
| 招聘人数 | value | int | 255 |

表 5-3 热门行业情况页面数据输入信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 长度 | 所属图表 |
| 岗位 | indus | Varchar | 255 | 实时行业  招聘热度图 |
| 招聘人数 | count | int | 255 |
| 招聘人数占比 | pp | double | 255 |
| 行业一 | Indu1 | varchar | 255 | 行业招聘  需求比例图 |
| 行业二 | Indus2 | varchar | 255 |
| 行业一占比 | P1 | double | 255 |
| 行业二占比 | P2 | double | 255 |
| 岗位 | exp | varchar | 255 | 热门行业  工资状况图 |
| 最高工资 | max | double | 255 |
| 最低工资 | min | double | 255 |

表 5-4 综合数据统计页面页面数据输入信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 长度 | 所属图表 |
| 国企 | guoqi | Varchar | 255 | 国企/民企工资  区间人数状况图 |
| 民企 | minqi | varchar | 255 |
| 学历 | exp | varchar | 255 | 工资  学历状况图 |
| 最高工资 | max | double | 255 |
| 最低工资 | min | double | 255 |
| 工作经验 | exp | varchar | 255 | 工作经验  需求状况图 |
| 招聘人数 | con | int | 255 |
| 工作经验 | exp | varchar | 255 | 工作经验  与工资状况图 |
| 最高工资 | max | double | 255 |
| 最低工资 | min | double | 255 |

表 5-5 相关信息分析页面数据输入信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 长度 | 所属图表 |
| 行业 | indus | varchar | 255 | 热门行业  招聘需求 |
| 招聘人数 | number | int | 255 |
| 行业 | indus | varchar | 255 | 热门行业  待遇等级 |
| 最低工资 | min | double | 255 |
| 最高工资 | max | double | 255 |
| 平均工资 | avg | double | 255 |
| 待遇等级 | level | double | 255 |
| 学历 | edu | varchar | 255 | 各学历  人数占比 |
| 招聘人数 | number | int | 255 |
| 学历 | data\_edu | varchar | 255 | 学历与  工资关系图 |
| 最高工资 | max | double | 255 |
| 最低工资 | min | double | 255 |

表 5-6 热门城市情况分析页面数据输入信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 长度 | 所属图表 |
| 城市 | name | varchar | 255 | 热门城市  招聘需求 |
| 招聘人数 | value | int | 255 |
| 城市 | city | varchar | 255 | 热门城市  工资情况图 |
| 最高工资 | max | double | 255 |
| 平均工资 | avg | double | 255 |
| 最低工资 | min | double | 255 |
| 城市 | city | varchar | 255 | 热门城市  招聘数据 |
| 招聘人数 | con | int | 255 |
| 标准差 | std | int | 255 |
| 平均工资 | avg | double | 255 |
| 城市 | city | varchar | 255 | 部分城市  工作满意度 |
| 工作满意度 | level | double | 255 |

表 5-7 趋势预测模块页面数据输入信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 长度 | 所属图表 |
| 工资 | sal | double | 255 | 2019年热门城市就业推荐指数 |
| 城市 | city | varchar | 255 |
| 推荐指数 | level | double | 255 |
| 行业 | indus | varchar | 255 | 2019年行业信息预测表 |
| 招聘人数 | con | int | 255 |
| 工资 | sal | double | 255 |
| 推荐等级 | level | int | 255 |
| 校园招聘 | citynum | int | 255 | 2019年各招聘渠道比例预测 |
| 企业招聘 | compnum | int | 255 |
| 学历 | edu | varchar | 255 | 2019年各学历层次工资预测 |
| 平均工资 | avgsal | double | 255 |
| 最高工资 | maxsal | double | 255 |
| 最低工资 | minsal | double | 255 |