

学号： 201571389

密级：           

**长江大学**

**硕 士 研 究 生 学 位 论 文**

**基于 python 的 Web 数据挖掘技术  
研究与实现**

专    业： 计算机科学与技术

研究方向： 数据挖掘

研 究 生： 刘熠

指导教师： 梁少华 副教授

论文起止日期： 2017 年 4 月至 2018 年 5 月

学号： 201571389

密级：                     



**长江大学**

**硕 士 研 究 生 学 位 论 文**

**基于 python 的 Web 数据挖掘技术  
研究与实现**

专    业： 计算机科学与技术

研究方向： 数据挖掘

研 究 生： 刘熠

指导教师： 梁少华 副教授

论文起止日期： 2017 年 4 月至 2018 年 5 月

# Research and Implementation on Web Data Mining Technology Based on Python

Major: Computer science and Technology  
Direction of Study: Data Mining  
Graduate Student: LiuYi  
Supervisor: Prof. LiangShaoHua

School of Computer Science  
Yangtze University  
April,2017to May,2018

## 摘要

随着互联网的迅速发展，大数据时代的来临，数据挖掘在从海量数据中探查潜在的价值信息起到了重要的作用，成为当下热门的研究和实践方向之一。python 作为数据挖掘领域中较为热门的程序语言，其丰富的技术库和强大的科学计算能力成为数据挖掘过程中不可或缺的工具。本次研究主要是基于 python 语言对智联招聘网的数据进行数据挖掘分析和建模，进而得出招聘信息薪资待遇预测分类模型。

本次研究主要分为如下步骤：数据源选择、数据采集、数据存储、数据预处理、数据建模和模型评估，通过算法构建了近邻和决策树两种分类模型，其次对两种模型的混淆矩阵数据进行计算，比较模型的预测准确率，最终得出准确率较高的数据模型。

本次研究所得的分类模型，可以帮助被招聘者在浏览网站招聘信息时预测薪资待遇水平，有效的评估招聘内容是否合适，以及对招聘岗位提供的薪资待遇是否感到满意，进而有效的提高求职者在寻求招聘岗位时的效率。

同时，该模型对于企业优化招聘信息也起到反馈性作用。企业可以根据模型对市场中有招聘信息的薪资水平进行有效分类，从而得到市场不同岗位的薪资水平分布现状，进而对企业招聘内容进行合理的调整，提高企业招聘效率，节约招聘成本，优化企业人才结构，提高企业在行业的竞争力。

**关键词：** 数据挖掘、python、分类算法、scrapy 网络爬虫

## Abstract

With the rapid development of the Internet and the advent of the era of big data, data mining has played an important role in exploring potential value information from massive data, and has become one of the hottest research and practice directions. As a hot programming language in the data mining field, python's rich technology library and powerful scientific computing capabilities have become indispensable tools in the data mining process. This research is mainly based on the Python language data mining and analysis of the data of Zhilian Recruitment Network, and then come up with a classification model for salary information of recruiting information.

This study is mainly divided into the following steps: data source selection, data acquisition, data storage, data preprocessing, data modeling, and model evaluation. Two classification models of neighbors and decision trees are built by using algorithms. The matrix data is calculated and compared with the prediction accuracy of the model. Finally, a data model with higher accuracy is obtained.

The classification model obtained in this study can help recruiters to predict the level of salary and benefits when browsing website recruitment information, effectively assess whether the recruitment content is appropriate, and whether they are satisfied with the salary and salary provided by the recruitment position, thereby effectively improving the efficient when seeking job opportunities.

At the same time, this model also plays a feedback role for enterprises to optimize recruitment information. The company can effectively classify the salary level of existing recruitment information in the market according to the model, so as to obtain the status quo of salary level distribution in different positions in the market, and then make reasonable adjustments to the content of enterprise recruitment, improve the efficiency of enterprise recruitment, save recruitment costs, optimize the talent structure of the enterprise and improve the competitiveness of enterprises in the industry.

**Keywords:** Data mining, python, classification algorithm, scrapy web crawler

# 目 录

摘 要 .....	I
Abstract.....	II
目 录 .....	III
第 1 章 绪论 .....	1
1.1 研究的背景及意义 .....	1
1.2 研究的目标 .....	1
1.3 研究内容与预期成果 .....	2
1.3.1 具体研究内容 .....	2
1.3.2 关键技术与解决思路 .....	2
1.3.3 研究环境 .....	3
1.3.4 预期研究成果 .....	4
1.4 文章的整体结构 .....	4
第 2 章 Web 数据挖掘原理 .....	5
2.1 Web 数据挖掘概述 .....	5
2.2 Web 数据挖掘分类 .....	5
2.3 Web 数据挖掘流程 .....	7
2.4 Web 数据挖掘常用技术 .....	8
2.5 小结 .....	8
第 3 章 Python 数据挖掘技术概述 .....	9
3.1 python 数据挖掘简介 .....	9
3.2 scrapy 爬虫框架 .....	9
3.2.1 网络爬虫 .....	9
3.2.2 scrapy 框架 .....	10
3.2.2 scrapy 爬虫运行过程 .....	11
3.3 数据挖掘常用 python 库 .....	12
3.3.1 pymongo 库介绍 .....	12
3.3.2 numpy 和 pandas 库介绍 .....	13
3.3.3 matplotlib 库介绍 .....	13
3.3.4 sklearn 库介绍 .....	13
3.4 小结 .....	14
第 4 章 招聘信息网数据挖掘方法研究 .....	15
4.1 数据采集阶段 .....	15
4.1.1 数据源目标选择 .....	15

4.1.2 数据采集及其规范 .....	16
4.2 scrapy 爬虫项目设计 .....	17
4.2.1 搭建 scrapy 工程 .....	17
4.2.2 网站反爬虫及应对措施 .....	17
4.3 数据预处理概述 .....	20
4.3.1 数据清洗方法 .....	21
4.3.2 数据规约方法 .....	22
4.3.3 数据变换方法 .....	22
4.4 数据分类算法 .....	23
4.4.1 KNN 算法 .....	23
4.4.2 决策树算法 .....	25
4.5 小结 .....	27
第 5 章 基于 python 的招聘信息网数据挖掘实现 .....	28
5.1 招聘信息采集 .....	28
5.1.1 爬取信息分析及实现 .....	28
5.1.2 数据格式设计与存储 .....	30
5.2 招聘信息数据预处理 .....	32
5.2.1 数据清洗实现 .....	32
5.2.2 数据变换 .....	35
5.2.3 新增特征值 .....	42
5.3 招聘信息数据挖掘建模 .....	44
5.3.1 基于决策树算法的建模 .....	44
5.3.2 基于 KNN 算法的建模 .....	45
5.4 模型评估 .....	45
5.4.1 决策树模型评估 .....	45
5.4.2 KNN 分类模型评估 .....	46
5.5 小结 .....	47
第 6 章 结束语 .....	48
6.1 论文成果与总结 .....	48
6.1.1 论文成果与创新点 .....	48
6.1.2 研究总结 .....	49
6.2 研究展望 .....	49
致谢 .....	50
参考文献 .....	51
个人简介 .....	56

# 第 1 章 绪论

## 1.1 研究的背景及意义

进入信息时代后，数据规模在不断的扩展，如何有效的从海量数据中提取所需信息成为当今各行各业关注的焦点。大数据时代，数据呈现出复杂、庞大的特性，一般的数据处理手段想从中获取到有价值的信息将十分困难。数据挖掘技术结合进化计算、信息论、信号处理等各多个领域的思想，通过多种复杂的算法，从大量未加工的数据集中解析出数据间潜在关系以及有效的知识信息。近年来，数据挖掘技术快速进步，在商务、医学、科学与工程等多种行业都取得了显著成果。数据挖掘技术随着大数据时代的到来，其研究价值也随之增高。

python 是一门面向对象的程序开发语言。强大的科学计算能力和丰富的代码库资源，使得 python 成为当今数据挖掘领域最为热门的开发工具之一。使用 python 进行数据挖掘不仅上手容易、门槛较低，运用其热门的 sklearn、pandas 以及 numpy 等工具库，将会大大降低数据挖掘流程中的多个环节工作量，从而使得研究者能够将更多精力投入到数据挖掘的设计和分析中，得出更为精准而有效的分析成果。本次研究，将有效利用 python 语言工具，进行 web 数据挖掘的研究与实现。

## 1.2 研究的目标

随着时代的发展，企业间相互竞争的模式已从产品竞争、质量竞争、技术竞争等方面慢慢演变成了各企业之间的人才竞争。在互联网飞速发展的时代大背景下，人才获取的途径已从线下识人逐渐向网络招聘的方向发展。《2017 年中国网络招聘行业半年度报告》<sup>[1]</sup>中指出 2017 年网络招聘求职者有近 1.6 亿人，2018 年预计将达到 1.78 亿人。近年来网络招聘一直呈现出上升趋势，其原因可以归结于网络招聘成本较低、招聘信息受众面广、招聘信息发布灵活可调节等。同时，通过网络平台，被招聘者和招聘者之间能第一时间有效的进行沟通互动，节省了大量时间成本。网络招聘已然逐渐成为各行各业招揽人才的主流途径之一。

在智联招聘、前程无忧等知名网络招聘网站中，每天都会发布大量各行各业的企业招聘信息。这些数据本身就具有很大的潜在挖掘价值。

本文的研究内容就是从这些招聘数据中着手，通过选定某一行业，并对其招聘信息进行大量数据采集与分析，构建一个该行业招聘信息的薪资待遇水平评估分类模型。因为招聘信息本身存在时效性，即不同时间段、不同行业引进人才的



薪资待遇标准会不断改变。根据人才招聘的普遍规律，可以初步假定模型有效性评估为半年时效，也就是说建立的该数据挖掘分类模型会在 2017 年 9 月至 2018 年 4 月期间具有最高可信度。因此，在论文数据准备初级阶段，将会采集相关有效数据，以供后续研究。

本次研究的目的是通过使用 python 语言工具，结合 python 语言的相关工具库资源，利用 Web 数据挖掘的原理，对招聘信息网的部分数据进行采集，最终将采集的数据进行挖掘，并对深圳的计算机互联网行业各职位招聘的薪资待遇情况进行预测分类，生成最终有效的分类模型。

## 1.3 研究内容与预期成果

### 1.3.1 具体研究内容

本论文将会结合 python 语言，对招聘信息网数据进行数据挖掘，实现一个可预测招聘信息薪资待遇水平的分类模型。本文具体工作内容如下：

1. 数据采集与存储。通过网络爬虫对招聘网站信息进行爬取，采集到所需数据后，根据特定数据格式写入数据库。
2. 数据预处理。对已存入数据库中的数据进行清洗，对空数据、无关联属性、无效数据等错误数据进行排除。在此阶段，会对数据新增薪资待遇特征值，该值为预测所需关键值。经过预处理阶段，将会生成最终有效的总数据集。
3. 数据挖掘建模。经过数据预处理得到真正有效的数据集。该阶段，将对数据集进行随机划分，得到训练集和测试集。根据训练集的数据，使用 pandas 提供的算法类生成器，建立分类模型。再利用划分的测试集数据对模型的有效性进行反复验证，最终获得有效的信息，得到预测分类模型。

### 1.3.2 关键技术与解决思路

#### （一）关键技术

在研究的过程中，将会使用 python 的相关技术。具体各流程的关键技术如下：

#### 1. 数据采集阶段关键技术。

在数据采集阶段主要使用了爬虫和数据库相关技术。本次研究将使用基于 python 开发的 scrapy 爬虫框架进行数据采集的主要工作，对于数据存储将使用 mongodb，利用其驱动 pymongo 进行数据转存等操作。

#### 2. 数据预处理阶段。

在数据预处理阶段，需要对数据进行批量的操作处理。python 的 pandas、numpy、matplotlib 工具提供了强大的矩阵运算操作功能。利用该工具，结合数

据预处理的相关知识，能快速大批量的进行数据预处理。

### 3. 数据建模

在数据建模阶段，主要是对数据预处理后的有效数据集进行划分处理。生成随机的训练集和测试集。在该阶段主要运用 python 的 sklearn 资源库，利用其随机分割的方法对数据集进行 8:2 的划分，生成合理的训练集和测试集。再通过 sklearn 的分类模型生成器生成 knn 分类模型和决策树模型。最后，测试分类模型的预测准确率，对两个模型进行对比，得到准确率较高的数据模型。

#### (二) 难点及解决方案

在此次挖掘的过程中，将会遇到了如下几项较为困难的问题：

1. 如何大量采集网页中的数据。网页数据对于网站本身存在着重要价值，在获取网页数据的过程中，网站服务器会对疑似爬虫的请求进行封锁，以此来保护网站本身的数据信息。由于网络反爬虫机制的存在，如何有效的回避反爬策略，成为本次试验将要突破的第一难点。解决方案：将会利用 ip 代理、延迟请求、仿浏览器请求等方式规避反爬虫规则。

2. 如何有效的对数据进行预处理。数据预处理对于数据挖掘而言极为重要，合理有效的预处理方式，将会对后续建模阶段有着重要的帮助；同时合理的预处理方法，可以有效的避免数据分类时的过拟合问题，提高数据建模的准确率和有效性。解决方案：合理制定预处理流程，从数据清洗、数据规约和数据变化三个方向去处理数据集。

3. 如何利用 python 实现分类算法，最终构建分类模型也是本次论文的难点之一。解决方案：理解分类算法原理，学习 sklearn 库的各算法分类器的文档，结合测试案例弄清如何构建分类模型。

### 1.3.3 研究环境

本次研究的环境主要如下表所示：

表 1-1 研究环境表

Table 1-1 The table of reserach enverionment.

开发平台	win10_64 位(专业版)
内存	8G
开发语言	python3.6.4
开发工具	pyCharm(professional 2017.3)
云服务器	腾讯云服务器(Ubuntu16.04.1 LTS 64 位)
数据库	mongodb v3.6.3

### 1.3.4 预期研究成果

本次研究通过爬虫获取数据，经过预处理后随机划分训练集和测试集。根据训练集构建分类模型，随后在使用划分的测试集对模型的准确性进行校验。

本次研究将会生成两个招聘信息薪资待遇预测分类模型，其中一个是基于 knn 近邻算法构建的分类模型，另一个是基于决策树算法构建的模型。通过各算法模型产生的混淆矩阵数据计算出模型的准确率。对比两组模型的预测准确率，最终可以得到较为精准的预测模型。该模型即为本次研究的预期成果。该成果可以对一定时间段内深圳的计算机互联网行业各职位招聘的薪资待遇情况进行预测分类，使用户在浏览招聘信息网时，有效的评估搜索的招聘信息是否提供了较好的薪资待遇，以此来更合理的去筛选心仪的招聘岗位；招聘企业也可以根据该结果判断企业在行业内的薪资待遇水平调整招聘内容，优化招聘信息，更为精准、快速的找到合适、优质人才，增加公司的人才竞争优势。

## 1.4 文章的整体结构

本文整体结构如下，共分 6 个章节：

第一章：绪论。主要叙述了本次研究的背景、目标、内容以及预期成果。

第二章：Web 数据挖掘原理。本章介绍了什么是 Web 数据挖掘、Web 数据挖掘的分类、Web 数据挖掘的流程以及 Web 数据挖掘的常用技术。

第三章：Python 数据挖掘技术概述。本章主要阐述了 python 数据挖掘简介、scrapy 爬虫框架、python 数据挖掘常用库。

第四章：招聘信息网数据挖掘方法研究。该章节主要对招聘信息网数据挖掘的流程进行阐述，如数据采集阶段、scrapy 爬虫设计、数据预处理流程以及分类算法等。

第五章：基于 python 的招聘信息网数据挖掘实现。本章是第四章数据挖掘流程的实现部分。包含招聘信息采集、招聘信息数据预处理、招聘信息数据挖掘建模和模型评估几个部分。

第六场：结束语。主要包括论文成果与总结和展望两个部分。

## 第 2 章 Web 数据挖掘原理

### 2.1 Web 数据挖掘概述

Web 数据挖掘最早起源于 1996 年，它也有如下的名称：Web 信息挖掘、Web 知识发现、网络信息挖掘等。

Web 数据挖掘(Web Data Mining)也被称为 Web 挖掘，其本质是数据挖掘技术与 Web 领域相结合的产物。该技术主要是从众多繁杂的网页信息中，抽取、规整出有价值的信息，通过数据转换、数据分析和数据建模处理，挖掘网页信息中的潜在信息，分析现有数据，根据数据呈现结果评估现状，甚至做出预测性判断，具有较大的商业和科研价值。

随着大数据的发展以及 Web 技术越来越普及，Web 信息的数据挖掘价值日渐显著。目前，Web 数据挖掘是已成为数据挖掘各领域中使用最为广泛的应用之一。通过 Web 挖掘可以更有效的分析用户行为、偏好等有价值的信息，后续再结合其他商业手段，为企业带来巨大的商业利益。

由于 Web 本身特性的原因，Web 数据挖掘呈现出如下特点：

1. 复杂性。Web 数据呈现多元化、复杂化。在互联网高速发展的背景下，各行各业数据呈现多元化、交叉化，Web 所含的数据信息也变得五花八门。数据存在形式也有所不同，如文本类型、视频类型、音频类型、图像类型等等。因此，如何快速、合理、准确的获取需求的信息，成为 Web 挖掘的首要问题。

2. 动态性。互联网 Web 技术的发展，数据来源变得越来越多样化。当今智能设备的大量普及，每日都会有大量信息不断涌入互联网。信息的更新迭代成为了 Web 信息中不可忽视的一个关键因素。数据的时效性也成为 Web 数据挖掘的难点之一。

3. 异构性。互联网形成初期，网站结构化没有统一的规范，这使得数据源之间产生了巨大差异。Web 数据挖掘中，需要解决从异构的网站结构中获取想要的信息数据的问题。

### 2.2 Web 数据挖掘分类

Web 数据挖掘的主要目的是从网页超链接、Web 使用日志以及网页内容中获取有用信息的过程。因此，根据 Web 挖掘中使用的数据类别的不同，可分为如下三类，如图：

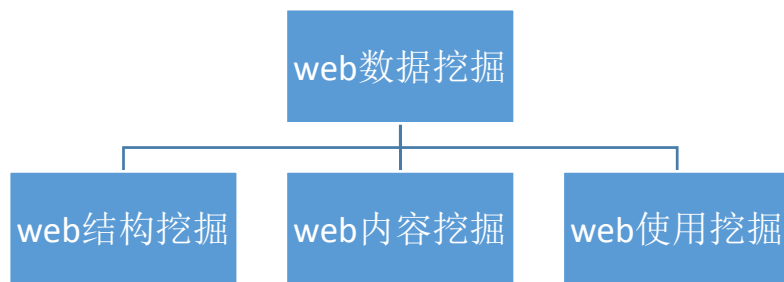


图 2-1 数据挖掘分类

Figure 2-1 The classification of Data Mining.

**Web 结构挖掘：**指的是通过网页包含的超链接中组织形成一组网站的结构化信息，根据这个结构层次获取有用的层级关系链，以此指导对页面进行分类和聚类，找到权威页面、中心页面，从而提高检索的性能。这与传统的数据挖掘有所不同，数据库表结构中没有型如超链接的数据结构。通过 Web 结构挖掘，可以有效的找出 Web 中重要的网页链接信息。因该分类方向不属于本文后续研究方向，所以不做过多展开。

**Web 使用挖掘：**在企业的各项商务活动推广中，通过数据挖掘集中获取用户点击行为日志，分析这些数据可以帮助理解用户隐藏在数据中的行为模式，做出预测性分析，从而改进站点的结构或为用户提供个性化的服务。同上，该分类方向不属于本文后续研究方向，所以不做过多展开。

**Web 内容挖掘：**根据 html 文档标签语义内容，从网页内容中识别并提取有效的信息和知识，例如：网页评论信息、商品简介描述等信息。在此基础上后续再使用数据挖掘算法，经过聚类 and 分类算法，抽取各数据的潜在信息，分析挖掘用户潜在需求，这也是不同于传统数据挖掘任务的地方。根据挖掘的数据内容格式又可细分为如下两类：

1. 文本挖掘。**Web 文本挖掘**是从 Web 文本文档中获取信息，并通过特征提取获得其隐藏模式的过程。在文本挖掘中，数据对象是一组结构化的 html 格式的文档信息。这样的数据结构本身不利于进行信息组织和信息获取。因此，将会对其进行一定的特征提取以方便获取需要的内容，并组合成结构化清晰的数据格式。在提取数据特征的过程中往往会遇到一个共同的难题，即数据本身存在多种维度的属性，如何选择有效的特征集降低数据属性维度成为 Web 文本挖掘的重要环节。目前特征提取的方法很多，常用的方法有如下：信息增益、特征熵、期望交叉熵、几率比等。其次在文本挖掘中数据库的选择和数据结构的设计也至关重要。如何高效的对采集的数据进行增删改查成为数据挖掘性能判断的一个标准之一。

2. 多媒体挖掘。**Web 多媒体数据挖掘**主要的对象是网络上的图像、音频和视频等数据的采集和挖掘分析。目前，Web 多媒体挖掘方法有如下：**a、**多媒体数据中的相似搜索；**b、**基于内容的检索系统和基于描述的检索系统；**c、**多媒体信息

数据的多维度分析；d、分类分析和预测分析；e、多媒体信息数据的关联规则挖掘。

### 2.3 Web 数据挖掘流程

在进行 Web 数据挖掘的过程中，可以大致的总结出其工作的主要流程。Web 数据挖掘的一般流程如下图所示：

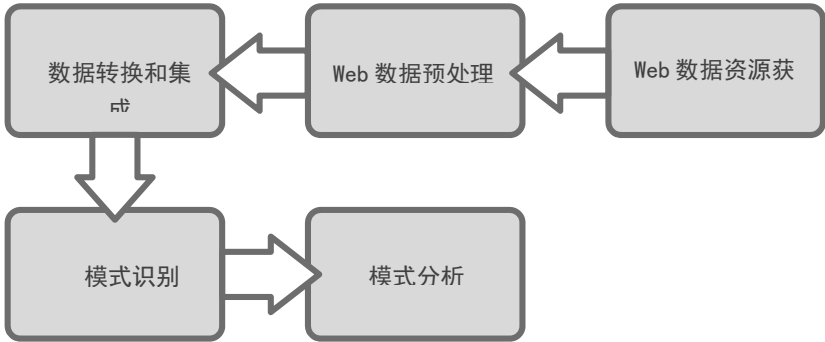


图 2-2 Web 数据挖掘一般流程

Figure 2-2 The general process of Web Data Mining.

**Web 数据资源获取：**该阶段是通过对 Web 网页结构内容的分析，获取网页中有用的内容数据的过程。该阶段中会采集到数据挖掘分析中所需的原始数据。该数据结构呈现多样化、复杂化，在该过程中还未对数据进行处理，数据存在大量冗余信息。

**Web 数据预处理：**该阶段将对获取到的数据进行下一步处理，通过有效的算法和分析方式对数据进行“修整”。主要包括数据清理、数据降噪、维规约、离散化等方式。通过预处理，数据质量将会有较大提高，有效的减少数据挖掘分析的时间，降低分析成本。

**数据转换和集成：**该阶段是将预处理后的数据进行格式化处理并存入数据库中待后续数据挖掘使用。根据设计好的数据仓库结构，将预处理后的数据载入数据库。此后将可以更加方便的对采集的数据进行增删改查的操作，提高了后续数据挖掘效率。

**模式识别：**此阶段已经拥有了采集好的结构化清晰的数据集。在此基础上运用分类算法、关联规则分析、聚类分析、统计分析等技术，对存储的 Web 数据进行数据挖掘，得出潜在的数据挖掘模式。

**模式分析：**该阶段使用现有的数据挖掘工具和挖掘技术进行下一步的模式分析，分析结果转为可视化的图表形式供分析人员更直观的解析挖掘结果，并对挖掘结果做出合理的阐述。

## 2.4 Web 数据挖掘常用技术

如上节所述，在 Web 数据挖掘的过程中会应用到相关的挖掘方法和技术。通过这些技术，可对数据进行有效的内容发掘。通过梳理，可以将常用的 Web 数据挖掘技术归结为如下几种：

1. 关联规则分析：关联规则即为型如  $X \rightarrow Y$  的表达式。关联规则分析的目的在于寻找数据间的隐藏的并发关系。一般分为两个阶段：第一，找寻并记录高频数据；第二，根据高频数据分析出数据间的内联关系。该技术是数据挖掘中的一项根本任务，也是数据挖掘领域被广泛使用的重要模型。

2. 分类分析：分类(classification)就是通过学习得到一个目标函数，把每个属性集映射到预先定义的类标号中的过程。该方法通过对数据库中的数据进行分析学习，得到有效的数据分类模型，根据学习建立的分析模型，挖掘出分类规则。通过该规则对该数据库中的其他数据进行分类规整。分类分析是数据挖掘中常用的方式之一，比较常用的模型如决策树模型、贝叶斯网络、支持向量机等。

3. 聚类分析：从技术和数据的角度讲，聚类分析就是通过数据统计学的原理对具有相似特性的数据进行特征提取，根据算法规则进行合理的划分属性类别。与分类分析所不同之处在于，在进行聚类分析时，对于数据本身的类别信息是不确定的，通过该算法可对新的数据进行规整。同时，聚类分析还可以被应用于其他算法的预处理阶段，即在聚类分析得出一定的划分结果后，将其他算法应用于处理结果数据的每个簇上。

4. 统计分析：Web 数据挖掘中较为常见的技术。其核心在于采集数据，通过不同的维度对收集的数据进行分析对比，最终得出结论。例如获取数据的某项属性的频数、某种记录的平均值等，可以通过这些统计记录对数据集进行基本的特征描述，从而得到最为直接的信息分析报告。

## 2.5 小结

综上所述，Web 数据挖掘其本质就是数据挖掘技术在 Web 领域的应用。目前国内外对于 Web 数据挖掘的研究主要从三个方向入手：Web 结构挖掘、Web 内容挖掘和 Web 使用挖掘。其本质是因为 Web 结构多变化，内容复杂性，使用广泛性等因素造成。

在实践中，大致分为如下五个阶段：(1) Web 资源获取阶段；(2) Web 数据预处理阶段；(3) 数据转换和集成阶段；(4) 模式识别阶段；(5) 模式分析阶段。同时文中也提到了在挖掘过程中常用的挖掘技术，例如分类分析、聚类分析和统计分析等，并对其做了简要的介绍。

## 第 3 章 Python 数据挖掘技术概述

### 3.1 python 数据挖掘简介

python 是当今最热门的程序设计脚本语言之一。它具有较好的解释性、交互性和面向对象等特性，使用 python 语言编写的程序代码具有较好的可读性，结构化的代码编写风格也使得程序开发和维护更具有高效性和可维护性。在对语言的学习投入上，python 具有较高的性价比，对于初学者而言可以迅速上手。python 语言具备的诸多特性使得其迅速成为风靡全球的热门语言之一。

python 语言的使用环境非常活跃。据报道，在 2017 年末热门开发语言的数据统计中，python 的使用占比已经超过 c 和 java 成为排名第一的热门语言。这一结果与 python 强大而又易用的语言特性分不开，也正是因为越来越多的人开始投入到 python 开发行列当中，使得 python 社区越来越完善，基于 python 开发的代码库也越来越丰富。

在近几年，数据挖掘技术已经多次成为计算机甚至其他行业的热门话题。而随着 python 语言技术和代码库的不断完善，数据挖掘技术已然和 python 语言成为了不可分割的一个整体。python 及其代码库提供了用户强大的科学计算能力，再加上不断开发出的支持数据挖掘各项技术的代码库，使得 python 成为数据挖掘中最为热门的工具语言之一。本次研究正是使用 python 的强大而又简易的多项功能，进行 Web 数据挖掘。

### 3.2 scrapy 爬虫框架

#### 3.2.1 网络爬虫

网络爬虫（spider）也叫网页蜘蛛，是通过设定好的规则，在网络上进行信息抓取的一组程序或者是一种脚本。随着网络的发展，各式各样的信息开始在网络上涌现。高效的网络数据搜索以及日益增长的大量自定向数据获取需求，使得爬虫已成为互联网时代各行各业的必修课。爬虫技术的兴起也催生了搜索引擎技术的不断前进。

本文研究的内容将涉及到网络爬虫技术。在数据挖掘的整个流程中，数据源即来自于众多结构复杂的 Web 网页中，下一章中将会设计爬虫程序的具体结构和内容，为后续的数据挖掘工作提取到数据集。



### 3.2.2 scrapy 框架

python 语言在网络爬虫设计中具有其独特的优势，scrapy 爬虫框架就是使用 python 语言开发并封装的一个强大的自动爬虫框架。这也是 python 语言的强大的社区库功能之一。无需再从零开始去设计爬虫框架，而是可以简单而高效的学习 python 的 scrapy 框架使用方法，通过 scrapy 框架提供的功能进行定向的数据爬取工作。在 scrapy 项目中，只需填入定制的爬虫规则，然后运行 scrapy，即可快速获得所需要的网页数据。

scrapy 强大的功能得益于他的构架，他总共有 8 个部分组成，也就是如下图所示的 8 个组件：

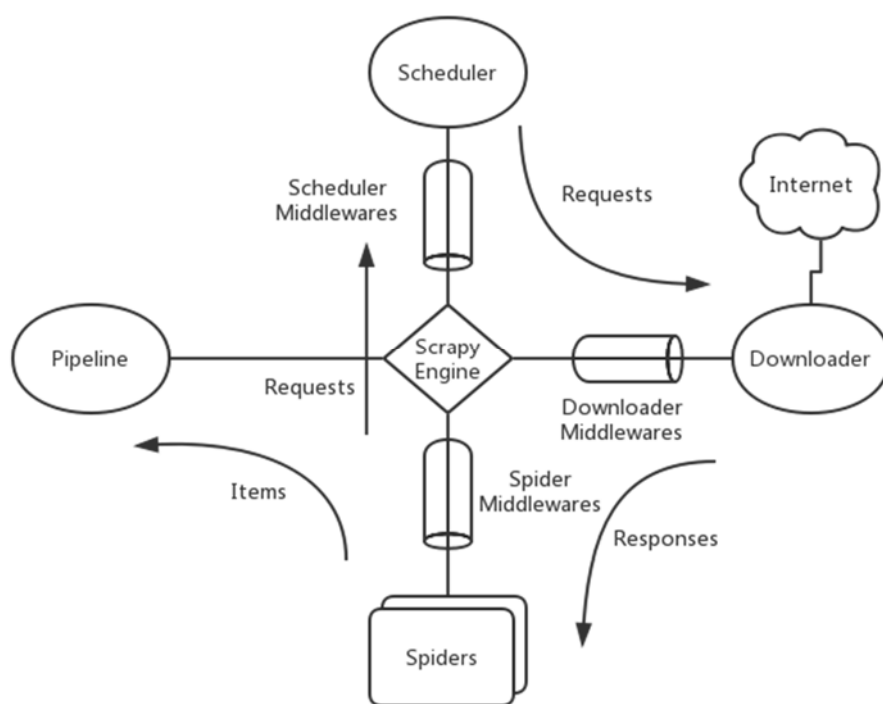


图 3-1 scrapy 构架

Figure 3-1 sScrapy structure.

1. scrapy Engine 组件：爬虫框架的引擎组件，负责所有组件的数据流动，并且在检测到相应的动作发生时，会触发特定事件。
2. Scheduler 组件：调度器组件，负责接收并创建请求队列，当引擎组件需要时为其提供请求队列，根据不同的 url 向网络请求数据。
3. Downloader 组件：下载器组件，根据调度器中接收的请求，负责下载网页数据，并将获取到响应数据推送给 spider 模块，进行后续处理。该组件采用异步请求的方式工作，其多线程的请求模式给数据请求阶段提供了高效的功能。

4. **spiders 组件**: 爬虫模块, 是整个 scrapy 框架的核心模块。其功能在于从特定的网页结构数据中获取指定信息, 在 scrapy 中被定义为实体(Item)。一个爬虫项目中可以拥有多个 spider 模块, 每个 spider 模块负责某个或者某些指定的网页。这样可以达到同时处理多个不同网页内容的需求。

5. **Pipeline 组件**: 数据通过 spiders 组件处理后流向管道组件(pipeline), pipeline 组件负责处理 spider 中提取出来的 item。在该阶段可以对数据进行清理、验证以及持久化(转存数据库)的处理。

6. **Downloader middlewares**: 下载器中间件是存在于 scrapy 引擎模块和下载器组件之间, 是两个组件之间的钩子(hook), 其功能是负责处理两个组件之间的请求体和响应体。经过其处理后再将数据传递给下个模块。

7. **spider middlewares**: spider 中间件位于 scrapy 引擎和 spiders 组件之间。负责处理爬虫数据的输入和输出。

8. **Scheduler middlewares**: 调度器中间件, 位于 scrapy 引擎和调度器之间, 其功能是处理引擎发送给网站或其他服务器的请求和响应。

可以看出, scrapy 各个组件负责不同的流程处理功能。scrapy 引擎负责整体组件调动。该框架结构设计和实现决定了框架的高效性、可扩展性以及鲁棒性。

### 3.2.2 scrapy 爬虫运行过程

scrapy 爬虫的流程如下图所示:

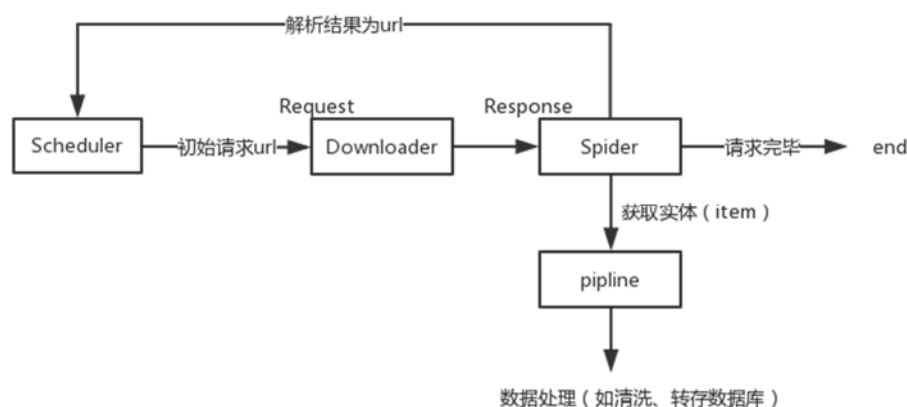


图 3-2 scrapy 爬虫流程

Figure 3-2 The process of scrapy crawl.

如上节所述, scrapy 爬虫的数据流动是 scrapy 引擎负责总调度控制。其过程描述如下:

1. 配置好 start\_urls, 规定爬取得初始目标, 完成后可以运行。

2. 引擎在接收到初始爬取的 url 目标地址后，通过调度器组件进行网络请求和响应的调度。
3. 下载器模块收到请求体，开始向互联网指定地址进行数据请求。并将最终生成的响应 Response 通过下载中间件发送给 scrapy 引擎。
4. 引擎接收到响应数据后通过 spider 中间件将数据传递给 spiders 模块，执行 spider 中定义好的回调函数，对数据体进行解析。
5. 数据体解析过程中获取到的信息实体（Item）将会继续推送到 pipeline 管道模块进行下一步处理。若解析数据为 url，则将该 url 通过 spider 中间件返给 scrapy 引擎，并通过引擎再次调用调度器对该 url 进行数据爬虫过程。
6. pipeline 组件接收到 spider 传递的实体后，可进行数据清洗、数据有效性验证、数据持久化等处理。该步骤已能获取到有用信息并可存入数据库中供下一步使用。
7. 经过不断循环的爬虫流程处理后，当 Response 响应体全部解析完毕，则数据请求流程执行完毕。后续 pipeline 数据处理完毕后，整个爬虫项目流程执行完成。

### 3.3 数据挖掘常用 python 库

采集完所需的数据集后，将进入到数据挖掘分析和建模阶段。在该阶段中，python 经过多年的整合与开发，已经具有了较为完善的包工具库，能够为接下来的数据挖掘阶段提供强大的技术支持和实现。本节将对后文使用到的关键 python 数据挖掘工具进行简要的概述。

#### 3.3.1 pymongo 库介绍

通过爬虫技术，可以在网站上获取到想要的信息。但是互联网数据本身存在异构性，处理各种不同的数据结构是一个较为复杂的过程。传统的关系型数据库，在存储不同网站不同复杂度的数据格式上有较大的难度。因此在数据爬虫中，更多的会选择 noSQL 非关系型数据库来存储想要的信息。

所谓非关系型数据库，不同于关系型数据库。noSQL 数据库可以存储各式各样的数据，同时具有高可扩展性和高可用性。数据模型更加灵活，数据读写更加简易，数据间无关系，方便数据扩展。当前热门的 noSQL 数据库中，选择 MongoDB 数据库来对爬取的数据进行存储，供后续数据挖掘操作使用。

pymongo 就是 MongoDB Driver 的 python 实现版本。他有效的搭建了 python 版本的连接 mongoDB 数据库的驱动环境。通过 pymongo 可以对本地或者远程服务器的 mongoDB 数据库进行链接，从而在获取到数据之后，将有用信息以文档的形式（mongoDB 数据库中，数据将以文档的形式存储）导入数据库。后续处理阶

段继续可以使用 pymongo 的环境对这些存入数据库中的数据进行增删改查操作。

不同于其他语言，pymongo 连接数据库，并操作数据库仅仅只需要短短几行代码即可。这也是 python 语言所带来的高封装性、高可用性的便利。如下所示：

```
import pymongo
client = pymongo.MongoClient('数据库地址及端口信息')
```

此时已经成功连接到 mongoDB 数据，在 client 实例中便可以随意操作数据库。

### 3.3.2 numpy 和 pandas 库介绍

numpy 是基于 python 开发的一种开源的支持多种数值运算的工具库。其最为突出的功能在于，该工具可以支持大规模的矩阵数据运算，且运算性能较高。numpy 提供了各种数学计算工具，可以实现如线性代数等强大的数学运算功能。

pandas 也是基于 python 开发的，他是在 numpy 的基础上进一步开发的一个数据分析工具包。他提供的一维数组（Series）和二维表格数据结构（DataFrame）是其核心数据模型。该数据包将常用的数据分析功能集合在一起，如通过 pandas 库，可以直接对二维数据表结构（DataFrame）进行简单的数据统计汇总，包括最值数据、平均数、正态分布情况等。在做数据预处理分析时，会结合 pandas 和 numpy 工具包，对数据集进行有效处理。根据统计计算的结果，通过清洗数据、处理缺失、消除噪声等操作，进一步提取出有价值的数据集。

numpy 和 pandas 也成为了 python 数据挖掘技术中的核心技术之一，对数据处理和分析阶段有着较为重要的作用。这也是 python 成为性能强大且高效的数据挖分析环境的重要原因之一。

### 3.3.3 matplotlib 库介绍

matplotlib 是 python 中最热门的绘图库之一。在数据预处理和分析阶段，有效的图形界面可以大幅度减少数据预处理所需要的时间，高效的了解数据集所反应出的信息结构分布。结合 python 的 numpy 和 pandas 的数据结构，matplotlib 工具包，可以对其数据结构进行图像可视化的展现。

利用 matplotlib 库，通过简单的代码处理，可以实现如散点图、折线图、直方图等多种数据分析图形。如此即通过简单的操作实现可视化数据分析。matplotlib 也成为 python 数据挖掘的常用工具之一。

### 3.3.4 sklearn 库介绍

基于 python 编写的 sklearn 库是数据挖掘中的核心工具库，在机器学习的领域里具有很高的知名度。该库中包含了很多数据挖掘中的算法，直接提供了该算法的使用接口。同时库中也含有测试和调试工具，供用户使用，以便对算法所需的

参数进行优化和调整。

sklearn 中封装了许多机器学习的方法，如下图：

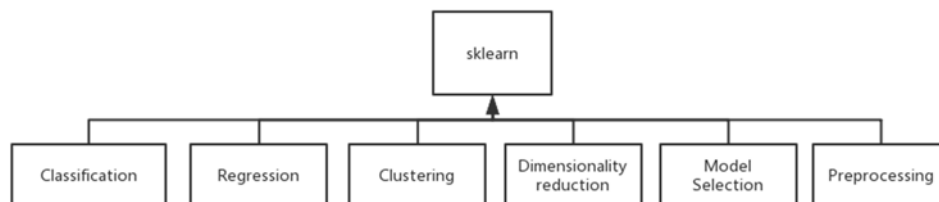


图 3-3 sklearn 机器学习方法

Figure 3-3 The method of machine learning by sklearn.

sklearn 工具包中封装了分类算法、回归算法、非监督分类算法、降维算法、模型选择和预处理功能。在下一章中，将会对具体使用的算法进行详述，这里就不在过多解释。

### 3.4 小结

本章节主要是对 python 在数据挖掘中的核心技术进行了简单介绍。数据采集阶段将使用 scrapy 爬虫框架来处理复杂的网络信息。爬取到的数据将会通过 pymongo 导入 MongoDB 数据库中，以便后续分析使用。同时，也介绍了在数据挖掘分析阶段常用的 python 工具库包，包括 numpy、pandas、matplotlib、sklearn 这些高效而强大的开发工具。在下一章中将设计对招聘信息网数据挖掘的具体研究方法和流程，将会在实际开发中使用这些 python 数据挖掘常用的工具。

## 第 4 章 招聘信息网数据挖掘方法研究

数据挖掘的流程大致如下图所示。本章主要对各个流程中所使用的方法和技术进行详述，在下一章中将基于 python 技术实现该数据挖掘流程。

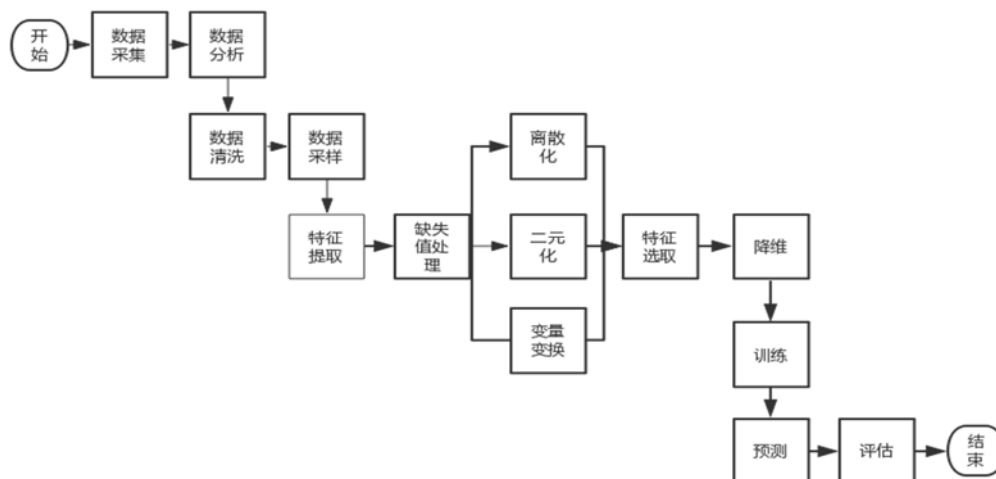


图 4-1 数据挖掘流程图

Figure 4-1 The process of Data Mining.

### 4.1 数据采集阶段

#### 4.1.1 数据源目标选择

2017 年 H1 中国网络招聘核心运营商网络招聘营收份额占比图如下：

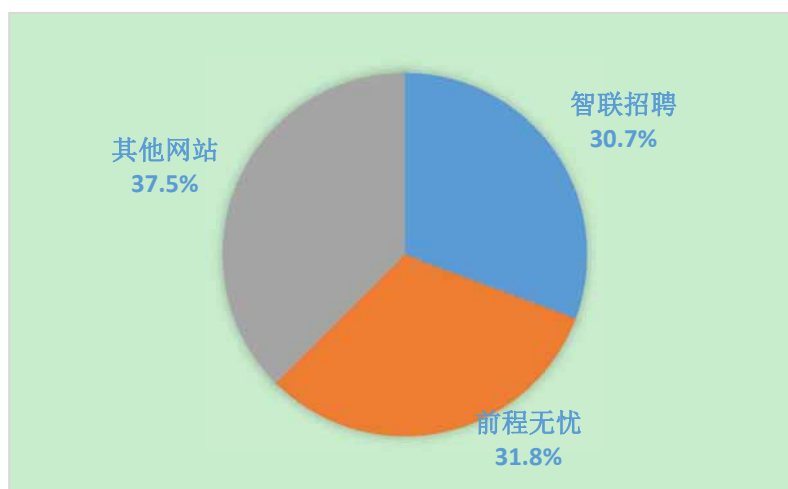


图 4-2 核心运营商占比

Figure 4-2 The proportion of core operators.

在网络招聘企业中，智联招聘市场份额占比 30.7%，前程无忧占有 31.8%，剩余其他网站占有 37.5% 的市场份额。尽管近些年不断有新的招聘网站涌入，但新招聘网站的成长尚需时间，未来市场依旧会持续双巨头的分布格局。本次研究将会对巨头之一的智联招聘网站进行数据采集工作，在此承诺采集到的数据将仅供学术研究，数据的所有权归属于智联招聘，不会将获取到的数据进行任何的商业用途。

#### 4.1.2 数据采集及其规范

招聘网站中数据呈现多元化、冗余信息较多的特性。在数据源选取目标数据时，需要经过仔细的筛选。数据的选择要具有如下几种特性：

唯一性。采集的数据要具有唯一性，数据初步筛选阶段即可对重复数据进行一定程度的过滤。例如，相同公司的相同岗位招聘信息可能存在多个时效的招聘数据，因此将会取最新招聘数据为主，覆盖原有的旧数据。以此来达到初步的数据唯一性规则。

准确性。采集的数据将会有多个维度，在数据采集阶段选取能准确描述数据的维度信息。若存在关联性较小或明显无关的维度信息，可以在该阶段去除。

相关性。采集的数据之间需要在某一维度或某几个维度上存在关联。

约定数据采集规范后，根据初步规范对网页数据进行定向采集。在智联网站中搜索招聘信息存在如下几个主要过滤条件：1、职位类别；2、行业类别；3、工作地点。通过以上三个过滤条件已能初步获取要指定职业类别的招聘信息数据。上述过滤条件中，工作地点固定为深圳，职位类别选定为软件/互联网/系统集成类型。而对于行业的筛选将会选择默认即全部行业。具体数据如下表：

表 4-1 搜索招聘信息所需过滤条件

Table 4-1 Search filters for recruitment information

数据过滤条件选择	
职位类别	软件/互联网/系统集成
行业类别	不限
工作地点	深圳

前文提到考虑数据的时效性，该数据的获取时间为 2017 年 9 月 20 日。本次研究将假定在获取数据的时间为该招聘数据的有效起始时间。则数据分析结果的有效性将从 2017 年 9 月 20 日起至 2018 年 4 月 20 日。在此期间，假定招聘市场对与软件/互联网/系统集成岗位的招聘待遇不会出现大幅度的变动。也就是在有效期之间，该数据集分析后的结果具有一定的准确性。

## 4.2 scrapy 爬虫项目设计

### 4.2.1 搭建 scrapy 工程

为了获取招聘数据，需要搭建 scrapy 爬虫项目，通过其强大的功能采集所需的网络数据。scrapy 的项目搭建极其简单，在确定好所需爬取数据的域名信息后，即可通过 scrapy 内置的脚手架命令自动完成项目框架构建工作。

在安装完成 scrapy 的工具包后，执行如下指令：

```
scrapy startproject recruit
```

运行上述命令后，会在所属目录下自动生成 recruit 项目文件夹，该项目的结构大致如下：

```
scrapy.cfg # 配置配置文件
recruit/   # 项目 python 模块
    __init__.py
    __pycache__
    items.py # Item 组件文件
    middlewares.py # middlewares 组件文件
    pipelines.py # pipelines 组件文件
    settings.py # 项目设置文件
    spiders/ # spiders 目录
        __init__.py
        __pycache__
```

在完成项目框架后，下一步则是生成自定义的爬虫文件。运行如下指令

```
scrapy genspider -t crawl zhaopin zhaopin.com
```

该指令中 -t crawl 表示根据通用爬虫模板的样式生成爬虫文件，同时爬虫的目标网站域名为 zhaopin.com。运行完指令后，在 spiders 文件中会生成 zhaopin.py 文件。该文件中会根据 crawl 模板生成自定的爬虫类 zhaopin。该类继承 scrapy.Spider 类的方法和属性。同时在允许爬取的域名属性中，会填入指定的 zhaopin.com 域名。

经过两条简单的命令，至此完成了所需的爬虫项目的大致项目结构搭建。这里可以充分的显现出 python 的 scrapy 框架的易用性、高封装性。使得爬虫项目变得更容易上手，减少重复而繁琐的前期工作，提高了开发效率。

### 4.2.2 网站反爬虫及应对措施

随着大数据的技术发展，人们开始越来越重视也更加需求数据本身的价值，网络中页面爬虫工具也开始日益增多。网页数据开始成为大家相互争夺的重要资



源之一。与此同时，网站也开始更加注重保护自己的数据资源，防止被竞争对手获取核心数据。反爬虫的关键目标在于阻止批量获取网站信息，导致网站服务器超负荷，同时也会流失大量供人免费查询的资源，致使网站本身减少了自身竞争力。

网站反爬虫主要通过以下几个方面：

1. 通过请求头部信息(Headers)反爬虫。在 http 请求的过程中，会带有该请求的头部信息。用户通过网页浏览器正常浏览的过程中，数据请求体的头部信息会带上该浏览器的标识信息。例如在 Headers 的 User-Agent 当中，显示的字段如下：Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/62.0.3202.75 Safari/537.36。该头部表明了，此次请求是通过 Mozilla 浏览器发起的。在例如 JavaScript 脚本中，通过编写批量的 ajax 请求可以获得网站信息内容，但该请求的头部信息往往与浏览器请求的不同。网站可以通过识别 User-Agent 头部信息，来对疑似爬虫的请求进行拦截，保护网站信息。

2. 通过分析用户行为来进行反爬虫。该方法主要通过检测用户的访问行为，例如用户所使用的 IP，在某一个短的时间范围内，多次访问请求了同一页面信息，或者改 IP 在短时间内存在着重复的请求的操作。这种异常的用户行为，会对服务器造成大量的负载，同时也具备爬虫的基本行为特征。网站可以在此分析上，对异常行为的用户进行拦截。

3. 上述两种异常情况多出现于静态的网站中，网站为了安全和反爬虫，可以使用动态页面的技术。网站本身的数据会通过 ajax 请求后获取，或者有 JavaScript 语言动态生成。在动态网页中，虽然其他人可以使用抓包软件对网络中的请求响应进行抓包，但是在获取的数据中，网站对 ajax 请求参数进行了加密，去除了参数的语义化和可读性，使得即使获取到信息，也很难解读其具体内容。同时对于网站的接口层做了高度的封装和加密，使得数据请求同一在内部调用，外界无法获取详情信息。

4. 网站登录验证。很多网站为了识别用户的真实性，即区分机器人脚本程序。在用户访问网站主体前，会需要进行登录验证。只有当成功登录后，才能正常的请求网站数据。

在建好的 scrapy 爬虫项目中，将会对网站的反爬虫策略进行有效的规避。通过上述网站主要的反爬策略内容，可以在 scrapy 的中间件模块(middlewares)中进行处理。对应于上述的几项反爬策略，在 middlewares.py 文件中将写入如下信息：

1. 为了通过网站头部对于 User-Agent 的识别检测，可以在对网站信息数据请求体中，通过头部信息伪装来规避反爬。同时，由于请求次数较多，可以再进一步的准备好一组浏览器头部列表，对后续每一次的请求随机的分配不同的请求头部，使得网站在识别爬虫项目的大批量请求时，会认为是从多个浏览器中发起的。

操作流程如下：

首先，引入伪装浏览器库。

```
import fake_useragent
```

其次，在自定义的中间件类中，为所有请求体设置头部信息。

```
class RecruitSpiderMiddleware ( object ):
    ...(省略)
    def process_request(self, request, spider):
    def get_ua():
        "Gets random UA based on the type setting (random, firefox...)"
        return getattr (self.ua, self.ua_type)
        user_agent_random = get_ua()
        request.headers.setdefault('User-Agent', user_agent_random)
    ...(省略)dd
```

在 setting.py 全局配置文件中引入该中间件：

```
DOWNLOADER_MIDDLEWARES = {
    'recruit.middlewares.RecruitSpiderMiddleware': 543,
    'scrapy.downloadermiddlewares.useragent.UserAgentMiddleware':None
}
```

通过这样的头部处理，达到了伪装浏览器请求的效果。使得在爬取数据的过程中，网站服务器对该请求会识别为浏览器的常规请求。

2. 为了避免高频请求带来的反爬虫处理，在 scrapy 爬虫项目中可以通过如下两个方面对爬虫项目进行优化，规避反扒策略。

(1) 使用代理 ip 进行请求数据。可以在采集一定量的免费可用的代理 ip，再次定义一个代理中间件，利用代理 ip 对数据源进行请求操作。在网站服务器对用户行为进行分析时，检测到的是多个不同 ip 的请求，则不会被识别为异常操作，以此来回避反爬虫策略。在设置代理 ip 的时候，会不断的在 ip 代理池中随机切换。以起到避免单 ip 重复请求的多次的情况。首先在 setting.py 的全局文件中配置好采集的 ip 代理池 PROXY\_IP\_POOL。随后在 middlewares.py 中写入如下代理中间件类：

```
import random
from scrapy.conf import settings
class RandomProxyMiddleware( object ):
    "动态设置 ip 代理"
    def __init ( self ):
        self.ip_pool = settings['PROXY_IP_POOL']
    def process_request (self, request, spider):
```

```
get_ip = random.choice( self.ip_pool )
request.meta["proxy"] = get_ip
```

# 对异常请求进行处理, 重选代理 ip 再请求

```
def process_exception ( self, request, exception, spider):
    get_ip = random.choice (self.ip_pool)
    request.meta["proxy"] = get_ip
    return request
```

接下来, 在 setting.py 全局配置文件中引入该中间件:

```
DOWNLOADER_MIDDLEWARES = {
```

```
    'recruit.middlewares.RandomProxyMiddleware': 542,
    'recruit.middlewares.RecruitSpiderMiddleware': 543,
    'scrapy.downloadermiddlewares.useragent.UserAgentMiddleware':None
}
```

(2) 设置请求频率, 提高多个请求的间隔时间以此来规避异常请求行为。由于设置了请求间隔, 势必会降低请求数据的效率, 可以通过设置多线程的请求策略, 来达到一定的优化效果。该操作在 scrapy 项目中已被高度封装, 可以在 setting.py 中进行全局设置。如下:

设置下载延迟时间: `DOWNLOAD_DELAY = 2`

设置最大当前请求数量: `CONCURRENT_REQUESTS = 8`

设置初始化请求延迟时间: `AUTOTHROTTLE_START_DELAY = 5`

设置最在请求延迟时间: `AUTOTHROTTLE_MAX_DELAY = 60`

(3) 对于动态请求数据的处理, 可以在抓取到请求 url 后反复尝试对比得出简单答案。例如, 在招聘信息的分页循环爬取过程中, 需要知道下一页的页面数据 url 的具体请求参数, 通过多次手动翻页, 比较 url 参数改变的情况, 以此得出翻页参数所代表的字段具体是什么。对于动态数据请求的处理, 可以通过这样反复查找规律, 对比操作后数据不同点来获取有用的信息, 以此来采集到需要的真是数据。

通过这些有效的反爬配置, 使得爬虫项目能够流畅的运行, 可以有效的规避网站服务器的封锁, 采集到需要的网站数据以供后续数据挖掘分析使用。

### 4.3 数据预处理概述

确定数据集后, 接下来的工作就是对数据集进行预处理。数据预处理就是对采集的数据进行加工, 通过合适的方法进行清洗、筛选、维规约等等。该阶段的目标是获得低纬度、低噪声、特征明确的高质量数据集。下面将对具体内容进行分析。

### 4.3.1 数据清洗方法

数据清洗的主要工作就是对采集的数据集进行检测和错误纠正。错误的、有误差的数据会降低数据集的质量。例如，由于处理数据请求体的信息时的错误，导致部分招聘信息中，薪资待遇出现了极低的个别数据体。这些数据的存在将会总体上拉低整个数据集的平均数值和期望，污染了数据集中其他数据的真实性。在数据清洗阶段就要将类似的问题数据筛选出来过滤掉，该阶段会从以下几个部分去对数据进行清洗：

#### 1. 数据采集的错误

数据采集错误主要是指在数据采集阶段，遗漏属性值的采集或其他因素采集到了与原定数据集不相关的数据。此过程只考虑常见的错误类型，比如采集数据是错误的输入信息导致数据集采集出错的情况。在此阶段检测中，需人工干预数据采集的过程，纠正一般错误。

#### 2. 离群点检测

离群点指的是不同于其他大多数正常数据的数据对象。在某个特征属性下，个别数据会出现偏离大多数据的现象，这种现象也可被称作为数据异常。衡量数据是否异常的标准有很多，甚至并不一定只有错误数据会被归为异常。有的正确数据在某种条件下也会被归为异常值，也就是离群点。例如，在对招聘信息进行离群点检测的过程中，会发现部分数据点出现大幅度偏离主体数据的情况。例如，检查后发现，该数据是某知名企业的高级人才招聘，相较于其他中小型企业而言，该名企对同层次岗位的薪资待遇有着明显的数倍差异。从现实的角度出发，该数据具有合理性，但对于此次数据挖掘目的而言，该数据可能存在大幅度偏离中心数据的情况，会对整体数据进行污染，在计算数据平均值等数据时会产生较明显的误差。因此，为了保证整体数据的准确性，可以有选择的人为判定该数据为异常数据，进行舍去。

#### 3. 遗漏值检测

采取的数据集中，单个数据可能会遗漏一个或多个属性数据。这种信息采集不完整的现象并不少见。在处理遗漏值的问题中通常有如下三种方式：

(1) 删除数据或该属性。这是最为简单的处理方式，当出现遗漏错误时删除掉遗漏的属性或者将整个数据体删除，这样可以确保留下的数据完整性。但是这种方式可能存在隐患，若删除的属性具有较高的分析价值，而数据集中遗漏改属性的数据只占极少部分，则会因此损失大量有用信息。

(2) 估计遗漏值。有些情况下，遗漏值可以通过预估来填补上。例如采集的招聘信息数据集中，有如下数据：

min edu	position	experience	com trade	com property	property	pay
大专	网页设计/制...	1年以下	互联网/电子...	民营	全职	4001-6000
大专	网页设计/制...	1年以下	互联网/电子...	民营	全职	

图 4-3 采集的招聘数据(部分)

Figure 4-3 The recruitment data (partial)

两条招聘信息对于学历、工作经验的要求相同，招聘的岗位都是网页设计全职。假设第二条招聘信息的薪资数据缺失，在此基础上可以合理的对数据进行预估，大致范围应该在 4000-6000 之间。若想更准确的估计，可以对整个数据集进行筛选计算出该职位在同等工作经验需求和学历需求下整个深圳市场中平均值，同时求出该行业的最低薪资和最高薪资作为预估的参考数据进行调整。

(3) 分析中忽略遗漏值。在做数据相似性比对时，可以选择忽略掉存在遗漏数据的属性，使用其他属性进行相似性分析。若数据本身的数据较多，寻找相似性的过程并不会受到明显的影响。但若数据集较小，数据属性较少的情况，这样的处理方式会造成较大的误差。

#### 4. 重复数据检测

数据集中可能包含重复的一组或多组数据，处理重复数据的关键点是要找出重复的原因。需要区分出是数据相似还是数据重复。若存在重复数据，可以进行合并操作，以此来保证数据的唯一性。

### 4.3.2 数据规约方法

数据规则主要指的是通过某种方法降低数据本身的属性维度。多维度的数据集分析中，会很容易带来维灾难。维灾难是一种在高维度下进行数据分析时产生的一种现象。一般在数据分析中，为了更好的描述数据分类的情况，会通过新增数据维度的方式来去重新精确定义数据，在一定的范围中这样的方式会给数据分析带来帮助。但随着数据维度的不断增加，当达到一定的值后，数据的高维度将会给数据分析带来巨大的困难。由于数据维度的增加，会使得数据在整体数据空间中所占有的比例越来越小。这样就意味着在数据量不足以去构建分类模型，同时高维度的情况会使得很多分类算法和聚类算法的精确度大幅度降低。

由此可以看出，数据规约的重要性。在预处理阶段，有效的降低数据维度，将会更好的提高后续算法的精准度，建立有效的数据模型。

由于本次试验的数据维度较低，且各属性数据信息具有较明确的分类特性。故而在后续实现的过程中，将简化数据规约流程。

### 4.3.3 数据变换方法

数据变换的过程主要包括如下三个部分：

### 1. 离散化

离散化通常在分类或者关联分析中会应用到，指的是将连续的属性进行离散化处理，转变为分类属性的过程。通常，不同算法使得离散化的结果也有所不同。离散化的操作主要包含两个任务：其一是确定分类的个数，其二则是确定连续属性和分类值的映射关系。

根据是否使用类信息，又可以将离散化的方法分为非监督离散化和监督离散化。

非监督离散化即不使用类信息的离散化方法。通常会使用等宽的方法将属性进行分割，但该方法会受离群点的影响。为了减少像离群点这样特殊情况所造成的影响，可以使用等频率划分或者等深划分的方法。监督离散化则会使用类信息来进行离散化处理，在分割区间时会附加上类的信息，这样可以更好的规避特殊点造成的影响，保证分类的稳定性和准确性。

### 2. 二元化

在很多数据挖掘算法中，特别是在分来算法中，对于数据连续或者离散的属性都会要求其转换成一个或两个属性。二元化的处理成为保证算法能有效进行的方法之一。简单的二元化处理如下：若数据的某一属性具有  $x$  个分类值，则可以在区间 $[0, x-1]$ 中选择一个整数赋值给某一个分类值中，这样就将分类值进行了整数化。随后，再将整数值转为二进制数完成二元化处理。二元化处理的好处在于算法运算中能更好的进行计算处理。

### 3. 变量变换

指的是用于变量的所有值变换。在对变量处理的时候，可以使用如绝对值、函数等方式对变量值进行变换。其目的在于调整变量值的呈现和分布形式。通过变换使得数据进一步规范化、清晰化的展现出来。例如，招聘信息数据中不同岗位的薪资待遇会有较大的差异，4000 与 40000 的差距则会有 10 倍之多。反应在视图上的结果为分布不均匀。可以通过单位换算或者等比例缩小等方式多数据进行一定比例的压缩，使得数据集分布更加集中，能更好的反映出数据的分类情况。

## 4.4 数据分类算法

本次数据挖掘所要达到的目的在于，通过分类分析，有效的判断出招聘信息所提出的薪资待遇高低的情况。为达到这一目标，将选用基于规则的决策树算法和基于距离的 KNN 算法进行分析预测。

### 4.4.1 KNN 算法

KNN 算法也叫 K 近邻算法，是数据挖掘中常用的分类算法之一，最早提出时

间是在 1968 年。该算法在机器学习领域经常会被提及，是一个较为成熟的非参数算法。本文主要是在分类问题中使用 K 近邻算法。

KNN 算法的思路是：通过给定的数据集进行训练建模，随后对于新的数据实例在模型中寻找最相邻的 k 个实例。在这 k 个实例中，存在最多的分类标签即被认定为该新实例的最终分类结果。K 近邻算法具有稳定、简单和高效的特点，但在大量数据集的分析中，该算法存在如下的缺陷：

1、K 值得选择会影响分类结果（图），如图：

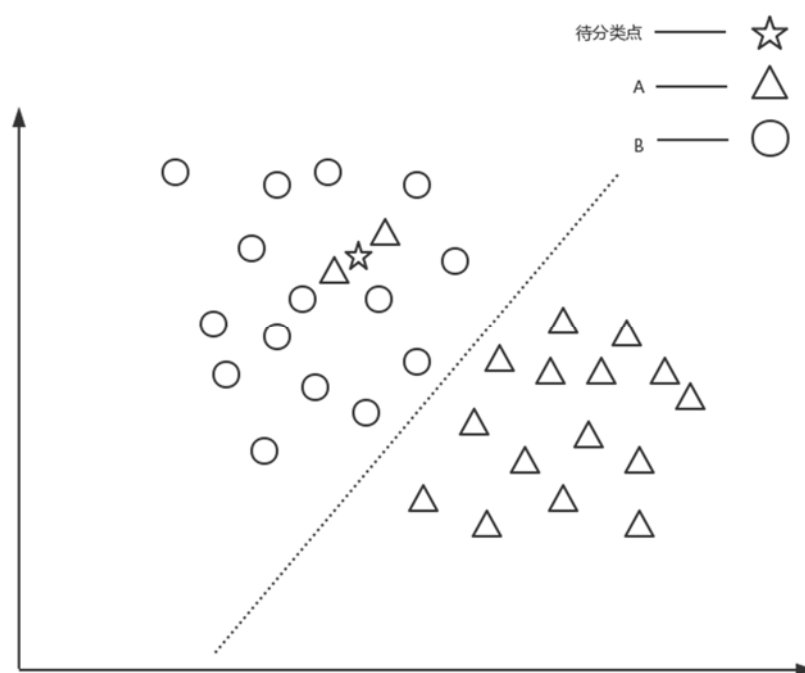


图 4-4 k 值的选择产生的误差

Figure 4-4 The error caused by the choice of k-value.

可以看到虽然从图中可以判断新实例所属的分类应该被归为 B，但是当 K 值为 3 的时候，最邻近新实例的点为 A(尽管新实例所在区域中 A 只存在 2 个，B 占大多数)，而当调整 K 值为 5 的时候，实例会被划分到 B。由此可以看出 K 值得选取会对分类的准确度造成一定的影响。在选择 K 值得时候应该进行多次试验测试，在多组模拟数据中进行反复试验，选择出一组准确率较高的作为 K 值选择的标准。

2、数据规模的大小和维度会直接影响算法的时间复杂度。K 近邻算法理论上可以对任意大小的数据集进行分类，但是，随着数据集的增加，数据维度的增多，计算这些数据个体间的距离会成几何倍数增长。因此，有效的选择训练集数据规模大小，以及适当降低数据维度会对算法的计算效率有着明显的作用。

3、距离计算的方式会影响分类的最终结果。常用的计算距离的方式有欧式距离、曼哈顿距离以及余弦距离。

欧式距离的特性：能够直观的反应数据在坐标系中的直线距离。本文也将采用欧式距离的计算方式来进行 K 近邻算法的距离计算。该算法存在的缺点是，若数据中存在大量的 0 值特征值，也就是说数据集属于稀疏矩阵，那么欧式距离算法将会不准确。因此在数据预处理阶段，需要对 0 值或者缺省值进行有效的处理，来规避这种情况。

曼哈顿距离的特性：曼哈顿距离又称为马氏距离，其算法为特征间的绝对轴距离之和。例如，A 的坐标为(x1, y1)，B 的坐标为(x2, y2)。则 AB 间的曼哈顿距离为 $|x1-x2| + |y1 - y2|$ 。曼哈顿算法的缺点在于，若数据集较大，数据密度高，该距离计算方式会掩盖数据其他特征间的邻近关系。

余弦距离的特性：预选距离指的是数据特征向量之间所成夹角的余弦值。该算法适合解决异常值较多的情况。但是该算法丢弃了向量的长度信息，在特定的场景下可能会造成数据信息流失的可能。

### 4.4.2 决策树算法

决策树(decision tree)是一种被广泛使用的分类技术。该算法是一种监督的学习算法。可以理解作为一种树状的流程图，其中位于上层的结点将决定下层结点的分布走向。

现有如下数据(取自于招聘网采集的真实数据)：

表 4-2 采集数据(部分)

Table 4-2 The collected data(particial)

职位	工作经验	学历	公司规模	薪资待遇(已处理)
Web 前端开发	1-3 年	大专	小	低
Web 前端开发	1-3 年	大专	中	正常
Web 前端开发	3-5 年	大专	小	正常
Web 前端开发	1-3 年	本科	小	正常
Web 前端开发	1-3 年	本科	大	高
Web 前端开发	1-3 年	大专	中	?

根据上述表格数据，绘制一个简单的决策树图形，用来对新的数据实例进行分类。(仅供描述决策树算法的思路，因此数据量选取很小，而且选择带有一定的定向性。对于最终生成的结果也存在过度拟合的现象。在后续实现过程中，将会导入更多的数据进行分析。)如下图所示：



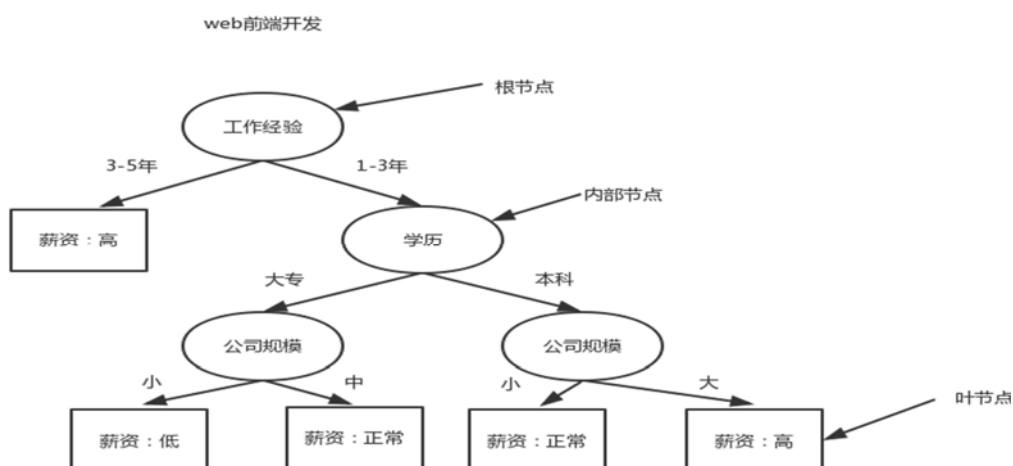


图 4-5 简化的决策树

Figure 4-5 Simplify decision tree.

树中包含了三种结点：

**根结点：**根结点的特征为没有输入的边，但是有零条或者有多条输出的边。它是整个决策时的入口。选择不同的根结点会呈现出不多的路径选择方式。

**内部结点：**例如学历属性的判断在此决策树中就是作为内部结点。它的特征是只有一条输入边，有一条或者多条输出边。

**叶结点：**也被称作终结点。特征为只有一条输入边，没有输出边。叶结点反应的最终分类结果。

将数据导入决策树模型会有如下情况：

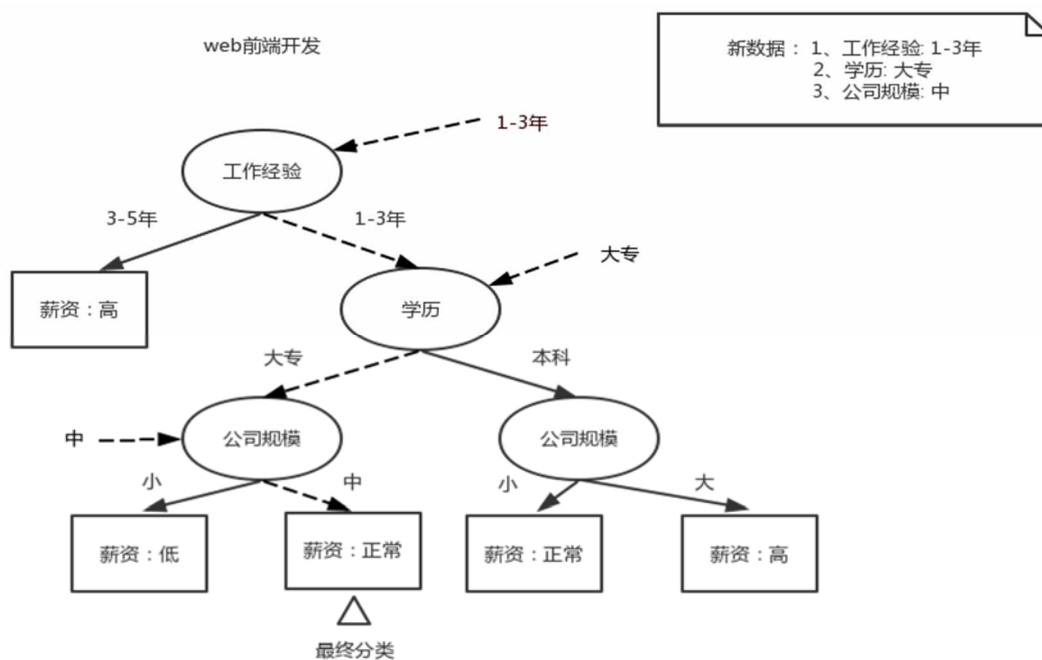


图 4-6 根据决策树对新数据进行分类

Figure 4-6 Classify new data according to decision trees.

待分类数据工作经验为 1-3 年，在树中第一次分类中将走向右侧的内部结点处；在对学历进行分类，新数据的学历为大专，走向左侧的内部结点。最后公司规模为中，则走向右侧的子结点。由于子结点反应的数据显示，该新数据可以被分类为薪资正常的类别。这样通过一个简单的数据和决策树模型，就对新的未分类的数据进行了预测。由于该数据是为了方便描述决策树算法原理而准备，本身就会具有很多数据过度拟合。在决策树的建立过程中经常会遇到类似问题，如何避免过度拟合的发生，将采用如下剪枝的策略进行回避。

剪枝是一个简化决策树过度拟合的过程。常用的剪枝方法有两种：

1、先剪枝。指的是通过停止树的构建过程从而达到剪枝的效果。当构建停止时，所有末结点将变为子结点。例如，设置决策树的高度，当决策树生长到这一高度时就停止，以达到剪枝的效果。也可以设置数据分类过程中实例集具有相同特征时停止。当一些实例经过不断分类到一定层级后，数据都具有相同的特征时即可停止分类。先剪枝的好处在于处理大量数据集时，不用构建完整的决策树模型即可完成分类。这样有利于提高算法效率，但是如何选择阈值非常困难，需要重复试验比对结果才能够找到合适值。

2、后剪枝。不同于先剪枝的处理方式，后剪枝将首先构造出完成的决策树模型，并允许树过度拟合训练数据。决策树构建完后，对整个模型进行评估。对置信度不够的内部节点，可以用出现频率最高的子结点进行替换，以此达到剪枝的效果。目前后剪枝的方式更为常用，而且实现难度较低。

## 4.5 小结

本章节主要是对招聘信息网数据挖掘流程进行设计，并对流程中的重要操作方法和核心算法进行了概述。

本次数据挖掘的主要流程分为四个步骤：1、数据采集阶段；2、scrapy 爬虫项目设计；3、数据预处理阶段；4、数据分类算法建模阶段。最后，对建立的数据模型进行效率评估。下一章，将会按照本章设计的流程，利用 python 工具实现整个数据挖掘工作，完成本次的试验目标。

在此流程中，核心的难点存在两处。分别是：(1) 数据预处理阶段，如何有效的对获取的数据进行预处理，提高数据集的有效性，大大提高建模阶段的数据成功率。同时也要考虑数据处理过程中的合理性，对数据变形的操作要能体现出适用性。因此，在下章会详细的对预处理的操作进行解析。(2) 数据分类算法。数据分类算法是后续数据建模的关键。本次将选择两种分类算法进行建模、评估。对于 KNN 算法和决策树算法的实现和操作也是本次挖掘的难点。

# 第 5 章 基于 python 的招聘信息网数据挖掘实现

## 5.1 招聘信息采集

根据上一章所设计阐述的挖掘流程，本章是进行数据挖掘具体实现过程的阐述。主要从爬取信息、数据预处理、数据建模三个方向去描述该流程的具体实现过程。

### 5.1.1 爬取信息分析及实现

上一章在数据采集的流程中已经设计好了所需的招聘信息搜索条件，即职业类别(软件/互联网/系统集成)、行业类别(不限)、工作地点(深圳)。但选定搜索条件后，会呈现如下的数据信息：

职位名称	反馈率	公司名称	职位月薪	工作地点	发布日期
<input type="checkbox"/> php项目经理  		深圳市云众信息科技有限公司 	8000-12000	深圳	置顶 
<input type="checkbox"/> 美工  		深圳市声朗乐器有限公司 	4001-6000	深圳	置顶 
<input type="checkbox"/> 2018春招岗位  		深圳市珍爱网信息技术有限公司  	面议	深圳	置顶 
<input type="checkbox"/> 美工/网页制作/平面设计 月薪5000-8000  		深圳玛丽温莎珠宝有限公司 	8000-10000	深圳-罗湖区	置顶 
<input type="checkbox"/> 淘宝美工  	90%	淘禧包装材料(深圳)有限公司 	6000-9500	深圳	置顶 

图 5-1 招聘条目信息(部分)

Figure 5-1 Recruitment entry information (partial)

php项目经理

深圳市云众信息科技有限公司 

五险一金

全勤奖

年终分红

股票期权

带薪年假

立即申请

职位月薪：8000-12000元/月 

工作地点：深圳

发布日期：招聘中

工作性质：全职

工作经验：5-10年

最低学历：本科

招聘人数：2人

职位类别：PHP开发工程师

深圳市云众信息科技有限公司 

公司规模：20-99人

公司性质：保密

公司行业：计算机软件

公司地址：深圳市坂田街道城市山海中心C615

职位描述

公司介绍

岗位职责：  
1、协助总经理商务洽谈、负责团队事务管理。  
2、主导网站、业务系统及技术改造类项目的系统分析与设计工作；  
3、重点技术难题攻关，保证业务系统安全、高效稳定运行；  
4、承担核心功能代码编写，核心模块维护，以及设计文档；  
5、持续系统性能优化，提升系统在大规模分布式系统环境下高并发大量的高处理性能；  
6、解决各类潜在系统技术风险，以及推动平台技术的革新；

小程序打开

举报

收藏

防诈骗提示

智联提示每一位求职者：若用人单位存在提供虚假信息、发布虚假招聘广告，以担保或者其他任何名义向求职者收取财物（如办卡费、押金、培训费），扣押或以保管为名索要身份证、毕业证及其他证件等行为，均涉嫌违法，请您提高警惕并注意保护个人信息！！

图 5-2 招聘详情

Figure 5-2 The recruitment details.

通过观察和分析，可以看到在招聘列表页将会获取到该招聘信息详情页的链接，点击链接进入详情页后将会得到该招聘条目的所有相关信息。在数据爬取阶

段，将所有招聘条目相关的有用的信息给予采集。组成需要的源数据集。

数据采集的环境及工具介绍:

表 5-1 数据采集环境及工具

Table 5-1 The Environment and Tools on Data Collection.

工具语言	python 3.6.4
爬虫框架	scrapy
运行平台	win10(64 位专业版)

在设计爬虫的过程中，最为麻烦的一个步骤就是在网页中分析元素节点，采集相关信息。该过程需要清晰的解读 html 的文件结构，能在复杂的 dom 结构中准确的定位所需要的 dom 节点所在位置以及需要的数据内容。scrapy 框架中的元素选择器 Xpath 功能可以较为方便的实现如上所述的功能。

xpath 是一种用来确定 XML 结构文档信息的一门路径语言。本次爬取 Web 数据的过程中，xpath 将读取 html 树状结构的信息，根据节点信息结构的不同主要分为三类：元素节点、属性节点和文本节点，xpath 根据指定好的路径信息在 html 的 DOM 结构中寻找相对应的节点。这样通过路径定位到需要的页面节点，通过此方式来获取节点有用的信息，再对信息进行整理即可获得所需的源数据。

通过审查页面源码，可以看到页面整体的 dom 结构信息。根据图中展示的信息，需要获取的有用信息有单条招聘详情页的具体链接、招聘条目名和详情页的相关数据。根据 Xpath 语法，可以将所需信息总结为如下：

表 5-2 采集属性所对应的 Xpath 语句

Table 5-2 The Xpath statement corresponding to the collection attribute.

招聘条目名	//div[@class="top-fixed-box"]/div/div/h1/text()
招聘条目详情链接	//td[@class="zwmc"]/div/a/@href
职位月薪	//div[@class="terminalpage-left"]/ul/li[1]/strong/text()
工作性质	//div[@class="terminalpage-left"]/ul/li[4]/strong/text()
工作经验	//div[@class="terminalpage-left"]/ul/li[5]/strong/text()
最低学历	//div[@class="terminalpage-left"]/ul/li[6]/strong/text()
招聘人数	//div[@class="terminalpage-left"]/ul/li[7]/strong/text()
招聘职位	//div[@class="terminalpage-left"]/ul/li[8]/strong/a/text()
公司规模	//div[@class="terminalpage-right"]/div/ul/li[1]/strong/text()
公司行业	//div[@class="terminalpage-right"]/div/ul/li[3]/strong/a/text()
公司性质	//div[@class="terminalpage-right"]/div/ul/li[2]/strong/text()

分析完所需的数据信息及如何解析节点信息后开始编写 scrapy 爬虫项目的 spider 模块。在自定义的/spiders/zhaopin.py 文件中完成爬虫解析的关键步骤。由于

自定义的 `zhaopin` 类继承至 `scrapy` 的 `Spider` 类, 因此该类中将定义爬取得初始动作, 以及后续是否跟进爬虫的逻辑判断。具体步骤如下:

1、选取爬取的初始页面, 在 `zhaopin` 类的 `start_urls` 属性中填入该初始页面的 `url` 值。

2、确定后续自动爬取的页数(经过测试该搜索信息有 91 页), 在 `__init__` 方法中将后续爬取的页数添加至实例中。经过测试发现, 翻页后 `url` 路径仅改变参数 `p` 数值, 在后续循环爬虫的过程中, 通过判断实例当前页码在修改 `p` 数值即可实现翻页后的数据爬取操作。

3、`parse` 方法中将会获取到指定 `url` 请求返回的数据, 在这个方法中将实现解析详情页连接, 并在存在有效链接的基础上继续请求详情页数据, 在该请求中会指定 `callback` 指定其返回数据的回调函数 `fetch_data`, 用于处理返回数据的解析操作。

4、在 `fetch_data` 中已获取到详情页的返回数据, 通过上面表格中所示的 `xpath` 路径, 在返回值 `response` 中提取所需的数据信息。经过简单的处理后将其绑定至定义的 `Item` 类属性中, 返回给 `pipelines` 组件进行下一步的流程处理。

5、在获取完所有页面以及页面中所有详情链接中的内容信息后, `spider` 组件工作流程则完成。后续将由 `pipelines` 模块进行数据格式重组, 以及写入数据库的操作。

### 5.1.2 数据格式设计与存储

通过 `spider` 模块, 已经将请求的数据返回给 `pipelines` 模块。在该模块中, 将会进行数据的整合, 以及连接数据库, 转存等操作。

根据上一节中对请求数据的处理, 单条请求的数据信息已存入 `Items` 自定义的属性中。本节将根据这些属性信息, 组织待存入数据库的数据基本结构。如下展示其 `json` 数据结构:

```
{
    "min_edu": "String",    # 最低学历
    "person": "String",    # 招聘人数
    "position": "String",   # 招聘职位
    "com_scale": "String",  # 公司规模
    "experience": "String", # 工作经验
    "name": "String",       # 招聘条目名
    "com_trade": "String",  # 公司规模
    "com_property": "String", # 公司性质
    "property": "String",   # 职位性质
}
```

```
"pay": "String"      # 职位月薪
}
```

确定完数据结构后，进行连接数据库操作。在模块中引入 pymongo 库，通过该库连接 mongodb 数据库，进行转存操作。pymongo 是基于 python 开发的 mongodb driver 驱动。可以通过简单的代码对远程数据库进行新增数据库，转存数据文档的操作。具体流程如下：

1. 引入 pymongo 库和全局配置

```
import pymongo
from scrapy.conf import settings
```

2. 在全局 setting.py 中配置好数据库连接所需的具体参数，如数据库 ip 地址及端口，数据库账号密码，以及数据库名和待操作的集合名(COLLECTION)

```
# setting.py
MONGO_HOST = "数据库 IP 地址"
MONGO_PORT = "27017"
MONGO_DB = "数据库名"
MONGO_COLL = "数据库集合名"
MONGO_USER = "数据库登录账号"
MONGO_PSW = "数据库登录密码"
```

3. 在 pipelines 的 process\_item 方法中，操作数据库，将格式化后的数据存入 mongodb 指定数据集中。

```
# pipelines.py
class RecruitPipeline ( object ):
    def __init__( self ):
        # 连接数据库
        self.client = pymongo.MongoClient ( host = settings["MONGO_HOST"],
port = settings["MONGO_PORT"])
        # 验证登录账号密码
        self.client.admin.authenticate ( settings["MONGO_USER"],
settings["MONGO_PSW"] )
        self.db = self.client[ settings["MONGO_DB"] ]
        self.coll = self.db[ settings["MONGO_COLL"] ]
    def process_item ( self, item, spider ):
        # 格式化获取的 item 数据
        ...
        # 存入数据库
```

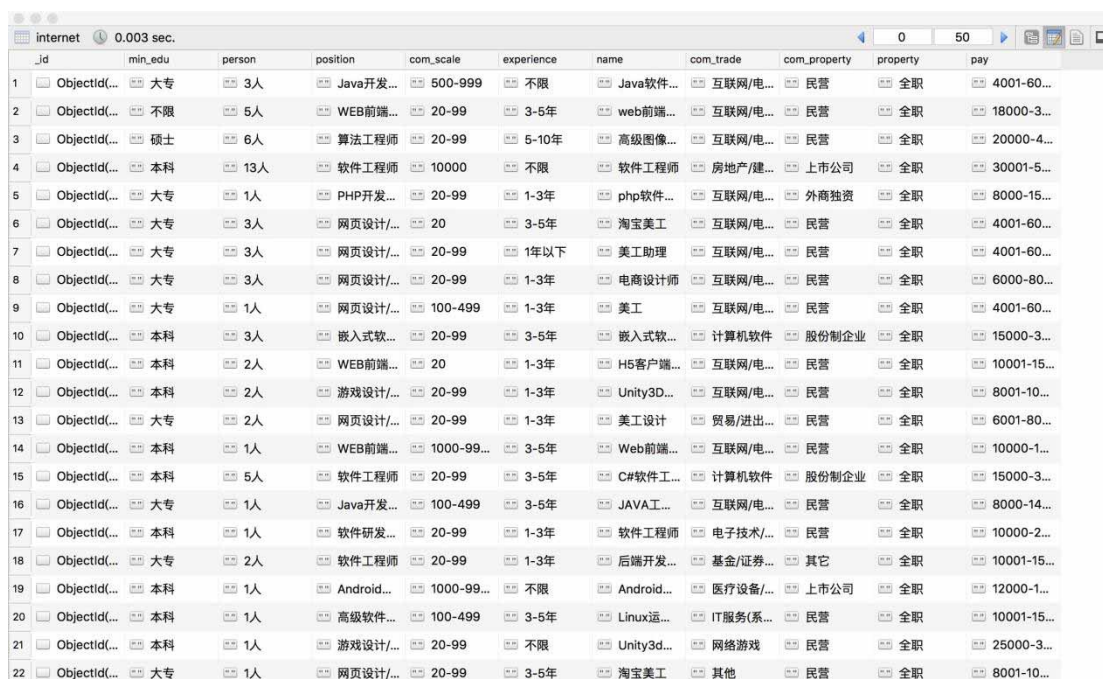
```
self.coll.insert(data)

return item
```

至此，整个爬虫项目的设置已全部完成。可以通过 scrapy 的运行指令开始进行数据采集操作。运行如下指令开始操作：

```
scrapy crawl recruit
```

本次数据采集耗时 5 小时 37 分钟，采集的数据集共有 5330 条。部分数据截图如下：



	_id	min_edu	person	position	com_scale	experience	name	com_trade	com_property	property	pay
1	Objectid(...)	大专	3人	Java开发...	500-999	不限	Java软件...	互联网/电...	民营	全职	4001-60...
2	Objectid(...)	不限	5人	WEB前端...	20-99	3-5年	web前端...	互联网/电...	民营	全职	18000-3...
3	Objectid(...)	硕士	6人	算法工程师	20-99	5-10年	高级图像...	互联网/电...	民营	全职	20000-4...
4	Objectid(...)	本科	13人	软件工程师	10000	不限	软件工程师	房地产/建...	上市公司	全职	30001-5...
5	Objectid(...)	大专	1人	PHP开发...	20-99	1-3年	php软件...	互联网/电...	外商独资	全职	8000-15...
6	Objectid(...)	大专	3人	网页设计/...	20	3-5年	淘宝美工	互联网/电...	民营	全职	4001-60...
7	Objectid(...)	大专	3人	网页设计/...	20-99	1年以下	美工助理	互联网/电...	民营	全职	4001-60...
8	Objectid(...)	大专	3人	网页设计/...	20-99	1-3年	电商设计师	互联网/电...	民营	全职	6000-80...
9	Objectid(...)	大专	1人	网页设计/...	100-499	1-3年	美工	互联网/电...	民营	全职	4001-60...
10	Objectid(...)	本科	3人	嵌入式软...	20-99	3-5年	嵌入式软...	计算机软件	股份制企业	全职	15000-3...
11	Objectid(...)	本科	2人	WEB前端...	20	1-3年	H5客户端...	互联网/电...	民营	全职	10001-15...
12	Objectid(...)	本科	2人	游戏设计/...	20-99	1-3年	Unity3D...	互联网/电...	民营	全职	8001-10...
13	Objectid(...)	大专	2人	网页设计/...	20-99	1-3年	美工设计	贸易/进出...	民营	全职	6001-80...
14	Objectid(...)	本科	1人	WEB前端...	1000-99...	3-5年	Web前端...	互联网/电...	民营	全职	10000-1...
15	Objectid(...)	本科	5人	软件工程师	20-99	3-5年	C#软件工...	计算机软件	股份制企业	全职	15000-3...
16	Objectid(...)	大专	1人	Java开发...	100-499	3-5年	JAVA工...	互联网/电...	民营	全职	8000-14...
17	Objectid(...)	本科	1人	软件研发...	20-99	1-3年	软件工程师	电子技术/...	民营	全职	10000-2...
18	Objectid(...)	大专	2人	软件工程师	20-99	1-3年	后端开发...	基金/证券...	其它	全职	10001-15...
19	Objectid(...)	本科	1人	Android...	1000-99...	不限	Android...	医疗设备/...	上市公司	全职	12000-1...
20	Objectid(...)	本科	1人	高级软件...	100-499	3-5年	Linux运...	IT服务(系...	民营	全职	10001-15...
21	Objectid(...)	本科	1人	游戏设计/...	20-99	不限	Unity3d...	网络游戏	民营	全职	25000-3...
22	Objectid(...)	大专	1人	网页设计/...	20-99	3-5年	淘宝美工	其他	民营	全职	8001-10...

图 5-3 mongodb 数据库存储的数据(部分)

Figure 5-3 The data stored in Mongodb database (part).

## 5.2 招聘信息数据预处理

### 5.2.1 数据清洗实现

上一章流程设计中已说明，在该阶段会从以下五个方面去对上节采集到的数据进行清洗，纠正采集到的错误数据。在清洗数据的过程中，将结合 python 的两个常用工具库 pandas 和 numpy。利用其封装的数据分析方法和计算能力，对数据进行清洗。

流程如下所示：

#### 1. 数据采集的错误

该阶段主要是检测采集数据的正确性。即单条数据属性是否为初始设计中的 10 项属性(由于 mongodb 数据库在数据导入的过程中会自动新增唯一表示 objectid，

因此在数据正确性检测中未将其算入总和。)对于数据的正确性检测，具体的流程如下所示：

第一步，操作 `mongodb` 导出所有的数据，导出格式为 `csv`。该步骤可以通过 `mongo shell` 的命令行语句实现：`mongoexport -h [数据库地址] -port 27017 -username [数据库账号] -password [数据库密码] -db [数据库名] -collection [数据集名] -type csv -o data.csv`

第二步，通过 `pandas` 的 `read_csv` 方法载入数据集赋予变量 `data`，以便后续分析。

第三步，判断数据采集的正确性。主要从以下几个方面

① 数据总数校验(数据行数校验)

```
print(data.count())
```

# 控制台输出结果为

```
RangeIndex(start=0, stop=5330, step=1)
```

显示结果为数据采集了 5330 条，与预计采集总数相同。校验成功。

② 数据列数校验

```
print(data.columns)
```

# 控制台输出结果为

```
Index(['Unnamed: 0', 'min_edu', 'person', 'position', 'com_scale', 'experience', 'name', 'com_trade', 'com_property', 'property', 'pay'], dtype='object')
```

可以看出采集的数据属性共有 10 项，符合原定采集的设想，校验成功。

通过检测发现，采集数据符合流程设计时的需求，为正确数据集。

2. 离群点检测

离群点检测的目的是排除偏离中心数据相对远的少数离群数据。去除该数据有利于提高建模数据的稳定性。本次数据分析的核心点在于对招聘信息的薪资情况分析。故而在检测离群数据时，将会取检测不同职位下薪资情况的分布，对该体系下的离群点进行适当的处理。具体流程如下：

第一步，获取所有的职位类别。根据 `data['position'].describe()` 可以查看到该列的数据分布情况。数据集中共有职位 33 种，由于每一种职位的分析过程相似，本文将只抽取频数最高的软件工程师职位数据进行详述。

第二步，获取职位为软件工程师的所有招聘数据，使用 `pandas` 数据结构 `DataFrame` 的 `value_counts` 方法可以统计出该职位的薪资分布数据。根据汇总结果得出，在这 1054 条招聘信息中，薪资总计有 89 种，其中频数仅为 1 的有 30 种共 30 条信息，在该类招聘信息集中占比约为 0.09%，为最小占比。此类数据具有较明显的单例性，无法反应数据集的普遍分布规律，因此认为存在明显的离群特性，将给予删除处理。因此最后将留下 1054 条数据，共有 59 种薪资分布情况。



对所有职位的薪资分布情况按第二步的方式进行检测排除，将数据占比低于0.1%的数据去除，完成离群点的检测和处理。

### 3. 遗漏值检测

该阶段对数据的遗漏情况进行检测，主要是判断是否存在空数据或者无法识别的数据。通过 `data.isnull().any()` 进行初步检测。结果如下：

min_edu	False
person	False
position	False
com_scale	False
experience	False
name	False
com_trade	False
com_property	False
property	False
pay	False

可以看出不存在空数据。

### 4. 重复数据检测

#### ① 检查重复行数据

重复数据会影响最终生成的模型分析结果，应该在数据清理阶段进行排除。利用 `pandas` 的 `uplicated` 方法能够检测数据中是否存在重复项，遍历生成的检测结果，当值为 `True` 的时候在控制台输出提示。操作如下：

```
dupdata = data.duplicated()
count = 0;
for item in dupdata:
    if(item):
        count += 1
print('total dup count:', count)
# 结果为 total dup count: 18
```

出现该结果的原因在于，在数据爬取的过程中出现了重复请求数据的操作。因此在该阶段，对重复数据进行删除。得到唯一的数据集。

#### ② 检查重复列信息

采集的数据中，共存在 10 个属性列。通过观察，发现招聘职业和招聘条目名存在着重复描述，同时招聘条目名不存在分类特性，该属性值无法有效的对数据进行分类处理。因此，去除掉招聘条目列信息。

### 5.2.2 数据变换

在采集的数据集信息中可以看到，数据数据值的显示呈现多样化，这样对后续建模的处理造成了一定的难度。本节需要对数据属性值进行相应的变换，方便后续建模的实现。根据前文所述，数据属性为 10 个。接下来会对如下属性进行分析和合适的变换操作。

#### 1、最低学历(min\_edu)

获取所有学历的信息，结果如下：

表 5-3 最低学历频数表

Table 5-3 The frequency of min\_edu.

最低学历	频数
不限	589
高中	10
中专	102
中技	22
大专	2103
本科	2438
硕士	63
博士	3

数据汇总结果可以看到，主要可以分成 4 种区间。对所有数据进行整合后归类为如下四类数值属性。

表 5-4 重映射的分类表

Table 5-4 The remapping classification.

原分类	映射数值属性	计数
不限 + 高中 + 中专 + 中技	0	723
大专	1	2103
本科	2	2438
硕士 + 博士	3	67

#### 2、招聘人数(person)

同上，获取招聘人数的统计信息：

表 5-5 招聘人数频数表

Table 5-5 The frequency of recruitment numbers.

招聘人数	频数
1 人	2445
2 人	1019
3 人	753
4 人	178
5 人	635
...	...
50 人	4
55 人	1
65 人	1
999 人	18
若干	24

统计结果显示在人数的划分中，存在大量的分类情况。会给后续建模的过程中增加大量的维度信息。通过表格数据，可以将低频数据进行合并分类，降低分类的维度。如下表所示：

表 5-6 重映射的人数分类表

Table5-6 The remapping classification of population.

新分类	映射数值属性	计数
1 人	1	2445
2 人	2	1019
3 人	3	753
4 人	4	178
5 人	5	635
6 人及以上	6	300

### 3、工作性质(property)

数据集中工作性质统计结果如下：

表 5-7 工作性质频数表

Table 5-7 The frequency of property.

工作性质	频数
全职	5048
实习	24
校园	5
兼职	4

根据表格可以看出，招聘的工作性质主要是全职。现将上述标量属性进行数值化映射，以方便后续的建模计算。映射表如下：

表 5-8 重映射的工作性质表

Table 5-8 The remapping classification of property.

工作性质	映射数值属性
全职	0
实习	1
校园	2
兼职	3

#### 4、公司性质(com\_property)

数据集中工作性质统计结果如下：

表 5-9 公司性质频数表

Table 5-9 The frequency of com\_property.

公司性质	频数
民营	3661
股份制企业	458
上市公司	295
合资	170
外商独资	169
国企	108
港澳台公司	19
事业单位	16
代表处	3
其他	130
保密	52

同上节，对公司属性进行数值映射，映射表如下：

表 5-10 重映射公司性质表

Table 5-10 The remapping classification of com\_property.

公司性质	映射数值属性
民营	0
股份制企业	1
上市公司	2
合资	3
外商独资	4
国企	5
港澳台公司	6
事业单位	7
代表处	8
其他	9
保密	10

#### 5、公司行业(com\_trade)

公司行业数据种类较多,通过 pandas 的统计描述功能得出共有 50 种不同的公司行业分类。同样,在数据处理阶段对该类标量值进行数值映射:

表 5-11 公司行业频数表及重映射属性

Table 5-11 The frequency of com\_trade.

公司行业	频数	映射数值属性
互联网/电子商务	1568	0
计算机软件	906	1
电子技术/半导体/集成电路	479	2
...		
政府/公共事业/非盈利机构	1	48
中介服务	1	49

#### 6、工作经验(experience)

同上，可对数据的工作经验值进行数值映射，映射表如下：

表 5-12 工作经验频数表及重映射值

Table 5-12 The remapping classification of experience.

工作经验	频数	映射数值属性
无经验	156	0
1 年以下	177	1
1-3 年	1534	2
3-5 年	1479	3
5-10 年	387	4
10 年以上	6	5
不限	1342	6

#### 7、招聘职位(position)

同上，可对数据的招聘职位进行数值映射，映射表如下：

表 5-13 招聘职位频数表及重映射值

Table 5-13 The frequency of position and remapping classification.

招聘职位	频数	映射数值属性
软件工程师	996	0
高级软件工程师	487	1
Java 开发工程师	407	2
...		
仿真应用工程师	9	30
计算机辅助设计师	7	31
高级硬件工程师	1	32

8、公司规模(com\_scale)

表 5-14 公司规模频数表

Table 5-14 The frequency of com\_scale.

公司规模	频数
20	256
20-99	1785
100-499	1938
500-999	442
1000-9999	699
10000	198
保密	12

为了提高分类效率，可以对公司规模范围进行如下的定义：

表 5-15 重映射公司规模区间表

Table 5-15 The remapping classification of com\_scale.

区间	映射数值属性	频数
100 以下	0	2041
500 以下	1	1938
10000 以下	2	1141
10000 以上	3	12

9、每月薪资

月薪资的数据是本次试验的主要分析信息之一，现数据集中薪资的属性值呈现为区间值，例如 5000-8000。通过统计得出，该区间值存在 178 种，这样的离散值在后续建模的过程中会创建出 178 中分类，使得数据的维度变得非常庞大，将大大降低建模的效率和性能。

首先，可以将原先的薪资范围字符串转换为有效的数值信息。例如，薪资为 5000-8000，在对其进行转换的过程中，可以取其中值来作为该招聘信息的最终月薪值。

然后，在数值转变完成后，对薪资进行再划分区间。此时薪资数据已转为数值型，具备重划分区间的基础条件。通过数据统计，得到如下数据：

表 5-16 月薪资特性表（单位元）

Table 5-16 The characteristics of pay.

最低薪资	1000
最高薪资	100000
平均薪资	12087

现将薪资以平均值为中介线，将数据分为两段 A 和 B。再将 A、B 区间划分为 5 个等步长的区间。最终将薪资范围划分为 10 个阶段。通过计算可以得出如下所示的最终划分区间。

表 5-17 A 区间薪资特性表（单位元）

Table 5-17 The characteristics of pay on A range.

A 区间	
最低薪资	1000
平均薪资	12087
步长	2217

表 5-18 B 区间薪资特性表（单位元）

Table 5-17 The characteristics of pay on B range.

B 区间	
最高薪资	100000
平均薪资	12087
步长	17582

表 5-19 最终薪资区间映射表

Table 5-19 The remapping classification of pay.

最终薪资划分区间	
1	1000 - 3217
2	3218 - 5434
3	5435 - 7651
4	7652 - 9868
5	9869 - 12085
6	12086 - 29667
7	29668 - 47249
8	47250 - 64831
9	64832 - 82413
10	82414 - 100000



通过划分，将薪资转为了 1-10 的范围区间值。有效的对数据进行了规整和分级。

### 5.2.3 新增特征值

本次试验的主要目的在于创建数据模型，用于评估招聘信息所提供岗位的薪资待遇情况。在本节，将新增特征属性招聘待遇(treatment)。该属性的具体值需要通过如下方法进行评估得到。

通过观察数据已经实际生活中的招聘情况了解，选定从两个方向进行评估：招聘职业和每月薪资。这两个属性所反映的是不同职业的薪资待遇详情。

同样以软件工程师职业为例，详述新增特征值的过程。

1. 筛选出 position 为软件工程师的所有数据。
2. 对筛选出的数据薪资分布进行汇总，找出分布规律。

表 5-20 软件工程师招聘信息薪资特性汇总

Table 5-20 The summary of software engineer's payment characteristic.

软件工程师招聘信息薪资情况汇总表	
数据总数	994
平均值	4.57
最小值	1
最大值	7

同时统计出该职业的薪资频数统计情况，如下表：

表 5-21 软件工程师招聘信息薪资情况汇总表

Table 5-21 The summary of software engineer's payment .

软件工程师招聘信息薪资情况汇总表	
pay 值	频数
1	8
2	112
3	159
4	151
5	153
6	408
7	3

3. 绘制条形图，最终对该职位的数据进行新增属性赋值。

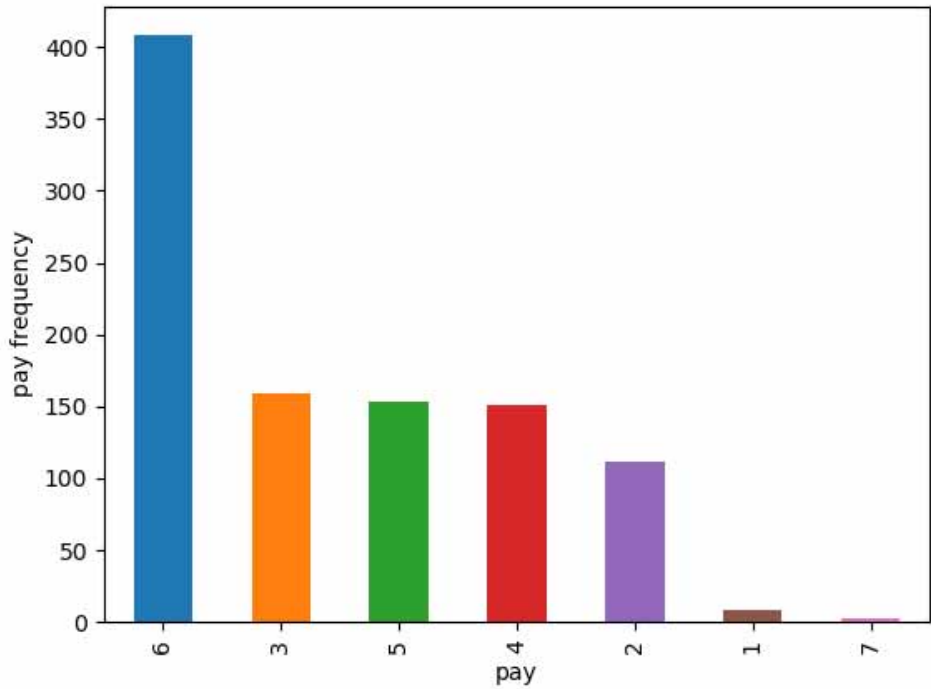


图 5-4 软件工程师招聘信息薪资频数条形图

Figure 5-4 The Bar Chart of software engineer's pament.

图像中可以直观的看出，薪资为 6 区间(12086 - 29667)的数据量最多，其余主要分布在 2、3、4、5 区间。最低频数在 1 和 7 区间。结合统计的平均值，可以将特征属性薪资待遇的值进行如下定义：

表 5-22 重映射薪资待遇情况划分表

Table 5-22 The remapping classification of treatment.

薪资待遇情况划分		
pay 值	招聘待遇	数值化
1 - 4	较低	0
5 - 6	合适	1
7	较高	2

经过分析，已对所有的软件工程师招聘信息新增了薪资待遇(treatment)这一新属性，并根据薪资的情况分别进行了较低、合适和较高三种标量赋值。随后对所

有的职位数据进行相应的赋值处理。至此，完成了整个数据预处理阶段的操作。

## 5.3 招聘信息数据挖掘建模

### 5.3.1 基于决策树算法的建模

使用 python 的 sklearn 库可以轻松实现决策树建模工作。在建模阶段，需要准备两种数据集：1、训练集；2、测试集。训练集用来生成最终的数据模型，而分离出的测试集用来对生成的决策树模型进行测试。如何生成有效的数据集是建模过程中的关键步骤，sklearn 库中 train\_test\_split 函数可实现该划分。具体操作流程如下：

```
from sklearn.tree import DecisionTreeClassifier
```

```
from sklearn.cross_validation import train_test_split
```

# data 为预处理后的数据集，经过数据预处理后，剩下的数据总数为 5069 条。训练集和测试集将在该数据集中分离出来。通过设置 test\_size 的值为 0.2，将数据集以 4:1 的比例划分为训练集和测试集（训练集:测试集 = 4:1）。

# knn 算法模型生成器

```
class knnGenerator:
def __init__(self, data):
    self.data = data
    def run ( self):
        x = np.array( self.data.drop ('treatment', axis = 1))
        y = np.array( self.data['treatment'])
    # 通过 train_test_split 生成训练集和测试集
        x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,
random_state=0)
```

此时 x\_train 和 y\_train 为生成的训练集，载入训练集后，通过 DecisionTreeClassifier 方法可以生成决策树。

```
clf = DecisionTreeClassifier(random_state=14, max_depth=5)
```

# 建立决策树模型

```
clf.fit( x_train, y_train)
```

通过上述流程，决策树模型已成功生成，随后将会使用测试数据进行验证。验证步骤将会在 5.4 节模型评估中进行详述。

### 5.3.2 基于 KNN 算法的建模

KNN 算法同决策树分类器生成过程相似。同样需要分离抽取两组数据集(训练集和测试集)。通过 sklearn 的 KNeighborsClassifier 方法进行 KNN 算法建模。具体流程如下：

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.cross_validation import train_test_split
class knnGenerator:
    def __init__(self, data):
        self.data = data
    def run(self):
        x = np.array(self.data.drop('treatment', axis=1))
        y = np.array(self.data['treatment'])

        x_train, x_test, y_train, y_test = train_test_split(
            x, y, test_size=0.2, random_state=0)
        neigh = KNeighborsClassifier(n_neighbors=5)
        # 建模
        neigh.fit(x_train, y_train)
```

通过上述流程，KNN 分类模型已成功生成，随后将会使用测试数据进行验证。验证步骤将会在 5.4 节模型评估中进行详述。

## 5.4 模型评估

在上一节中，已经通过数据集分离出的训练集生成了决策树分类模型和 KNN 分类模型。本节，将会通过各自分离出的测试集进行预测分析，同时进行数据准确性校验，通过校验将会得出两个模型的预测准确率。

### 5.4.1 决策树模型评估

在 5.3.1 中通过 sklearn 的 DecisionTreeClassifier 进行了决策树建模。在本节中，将会对 x\_test、y\_test 测试集进行预测操作。流程及运行结果如下：

```
# 预测
pre = neigh.predict(x_test)
```

pre 则为测试集生成的预测结果，可以通过混淆矩阵来对预测结果进行分析。sklearn 中已经对该方法进行了封装，使用以下代码生成混淆矩阵，对模型的分类结果进行评估。

# 使用 metrics 的 confusion\_matrix 生成混淆矩阵结果

```
metrics.confusion_matrix(y_test, pre)
```

结果如下表：

表 5-23 决策树模型混淆矩阵表

Table 5-23 The confusion matrix of Decision tree model.

预测值 实际值	0（较低）	1（合适）	2（较高）
0（较低）	358	11	0
1（合适）	1	570	0
2（较高）	7	12	55

通过混淆矩阵，可以得出模型的识别率和误分类率，如下表：

表 5-24 决策树模型准确率和错误率

Table 5-24 The accuracy and error rate of Decision tree model.

准确率	错误率
96.94%	3.06%

#### 5.4.2 KNN 分类模型评估

同上节步骤一样，通过测试集对模型进行预测，并分析预测的结果。流程如下所示：

```
# 预测
pre = neigh.predict(x_test)
# 生成混淆矩阵
metrics.confusion_matrix(y_test, pre)
```

结果如下表：

表 5-25 Knn 模型混淆矩阵表

Table 5-25 The confusion matrix of KNN model.

预测值	0（较低）	1（合适）	2（较高）

实际值			
0（较低）	312	41	16
1（合适）	11	556	4
2（较高）	18	21	35

通过混淆矩阵，可以得出模型的识别率和误分类率，如下表：

表 5-26 Knn 模型识别率和误分类率表

Table 5-26 The accuracy and error rate of KNN model.

准确率	错误率
89.05%	10.95%

## 5.5 小结

本章根据第四章的流程设计，使用 python 语言及其常用库实现了数据挖掘的全过程。包括数据采集、数据预处理、数据建模和模型评估操作。通过 sklearn 库，实现了决策树和 KNN 近邻算法的模型建立。并分别对采集的数据集进行了测试。通过测试结果生成的混淆矩阵呈现的数据，得出了各自的准确率和错误率。

其中，决策树模型准确率为 96.94%，错误率为 3.06%。KNN 分类模型的准确率为 89.05%，错误率为 10.95%。可以看出，决策树模型对于本次的数据挖掘结果预测的准确度更高。决策树模型的构建，将会是本次研究最终获得的结果。通过该模型，可以对满足过滤条件（深圳、互联网方向）的新的招聘信息薪资待遇水平进行有效的预测评估。用户可以知道自己查看的招聘信息，在该行业中是否有较为满意的薪资待遇，而企业则可以通过评估结果，判断公司在行业内的薪资水平，进而调整薪资标准，提高招聘效率。

## 第 6 章 结束语

### 6.1 论文成果与总结

#### 6.1.1 论文成果与创新点

##### (1) 论文成果

随着互联网的迅速发展，大数据时代的来临，数据挖掘成为了当下热门的研究和实践方向之一。python 作为数据挖掘领域中较为热门的程序语言，其丰富的技术库和强大的科学计算能力成为数据挖掘过程中不可或缺的工具。本次研究主要是基于 python 语言对智联招聘网的数据进行数据挖掘分析和建模，进而得出招聘信息薪资待遇预测分类模型。

本次研究主要分为如下步骤：数据源选择、数据采集、数据存储、数据预处理、数据建模和模型评估，通过算法构建了近邻和决策树两种分类模型，其次对两种模型的混淆矩阵数据进行计算，比较模型的预测准确率，最终得出准确率较高的是决策树模型。使用该模型，可以对智联招聘中互联网行业的招聘信息薪资待遇情况进行有效预测，通过测试集校验，模型的准确率可以达到 96.94%。

本次研究所得的决策树分类模型，可以帮助被招聘者在浏览网站招聘信息时预测薪资待遇水平，有效的评估招聘内容是否合适，以及对招聘岗位提供的薪资待遇是否感到满意，进而有效的提高求职者在寻求招聘岗位时的效率。

同时，该模型对于企业优化招聘信息也起到反馈性作用。企业可以根据模型对市场中有招聘信息薪资水平进行有效分类，从而得到市场不同岗位的薪资水平分布现状，进而对企业招聘内容进行合理的调整，提高企业招聘效率，节约招聘成本，优化企业人才结构，提高企业在行业的竞争力。

##### (2) 创新点

本次研究主要的是针对网络招聘信息中薪资待遇的情况进行了数据挖掘的工作，在研究过程中存在如下创新点：

1、基于 python 实现对网络招聘信息数据挖掘，新增特性值“待遇”，根据分类模型对招聘信息进行待遇值预测。本次数据挖掘主要采用 python 的开发环境进行，对数据挖掘过程中的数据采集、数据分析、数据处理以及数据建模均有实现。从本次研究的过程中可以看到 python 对于数据挖掘提供了强大的支持。

2、本次对网络招聘信息薪资的数据挖掘主要立足于求职者角度去分析，提出招聘待遇新特性，既可提高求职者招聘效率，又能一定程度减少企业招聘成本。具有一定商业用途。

### 6.1.2 研究总结

论文详述了此次研究的工作涉及的相关理论以及研究的内容和结果。

数据采集阶段，使用 python 的 scrapy 爬虫库，充分结合 Web 数据挖掘原理，在招聘信息网中采集到了试验所需的数据集。

数据预处理阶段，利用 python 的 pandas 强大的矩阵运算能力，结合数据预处理的各个方法，对数据集进行有效的处理，生成有最终有效的数据集合。

数据建模阶段，利用 python 的机器学习库 sklearn，通过简短的代码实现了决策和 KNN 算法的模型构建。

模型评估，同样适用 sklearn 的模型预测和混淆矩阵的计算方法，对测试集进行预测分类。根据预测结果生产混淆矩阵，计算出模型的准确率和错误率。对两个分类器进行了对比评估，得出决策树具有较高的预测准确度的结果，最终实现了较为理想的预测模型，也达到了本次试验的目的。

## 6.2 研究展望

本论文利用 python 语言对招聘信息网的部分数据进行了数据挖掘和分析，最终得出了较高准确度的基于决策树的算法模型。但本次数据挖掘实现过程主要是阐述如何利用 python 进行数据挖掘工作，以及对这数据挖掘流程进行了简单的实现，并未对最终生成的模型进行重复的参数调整和模型优化等细节操作。在本论文的基础上可以继续进行如下方面的研究工作：

1. 更细化的研究招聘薪资待遇的划分类别。由于本论文的研究方向主要是结合 python 对招聘数据的挖掘进行实践，注重点在于挖掘流程本身，并未对待遇划分进行更为细致精确的计算。本文中待遇划分只是结合了两种特征值进行估算的结果，后续研究可以考虑新增多个维度去衡量。

2. 由于数据源属性的限制，影响数据分类的某些因素未被采集进来。例如招聘信息公司的具体情况、招聘信息中简介部分的某些信息会对薪资的划分产生影响。

3. 本次数据源的规模大小使得研究结果具有一定局限性，同时招聘数据仅仅只是计算机行业某一分类的数据信息，可以采集更多行业的数据进行分析，得出更为真实准确的结果。



## 致谢

非常感谢在读研期间给予我帮助和指导的老师和同学，是你们让我在这三年里成长了很多，收获了很多。最要感谢的是我的导师梁少华老师，论文的选题、研究内容、组织结构、系统的研究进展以及各个阶段的研究方向都离不开梁老师的悉心指导，老师的严谨治学、孜孜不倦的研究态度我将一直谨记于心。

三年的学习生涯即将结束，我十分明白，这三年里带给我的不仅仅是知识的增长，更多的是精神和认知的提高，让我懂得更多，看得更远。这段时间里学习和收获的知识将会伴随我一生，受益一生。

最后还要感谢我的父母，是他们给予我无限的鼓励和信任，让我有信心走完三年的求学时光。在此，我要感谢所有对我有过帮助和鼓励的老师、同学和家人，祝老师们身体健康、事业顺心；祝同窗心想事成、前程似锦；祝家人和和美美、幸福安康！

## 参考文献

- [1] (挪)Magnus Lie Hetland 著, 司维, 曾军巍, 谭颖华译. Python 基础教程[M]. 北京: 人民邮电出版社, 2010. 7
- [2] (美)Pang-Ning Tan, Michael Steinbach, Vipin Kumar 著, 范明, 范宏建译. 数据挖掘导论[M]. 北京: 人民邮电出版社, 2011.
- [3] (美)Paul Barry 著, 林琪, 郭静译. 深入浅出 Python (中文版) [M]. 北京: 中国电力出版社, 2012.
- [4] (美)韩家炜等著, 范明等译. 数据挖掘概念与技术[M]. 北京: 机械工业出版社, 2012.
- [5] (加)Toby Donaldson 著, 袁国忠译. Python 编程入门[M]. 北京: 人民邮电出版社, 2013
- [6] (美)刘兵著, 俞勇译. Web 数据挖掘[M]. 北京: 清华大学出版社, 2013.
- [7] 涂子沛著. 数据之巅[M]. 北京: 中信出版社, 2014.
- [8] 麦金尼(Wes McKinney)著, 唐学韬等译. 利用 Python 进行数据分析[M]. 北京: 机械工业出版社, 2014
- [9] (美)萨默菲尔德著, 王弘博, 孙传庆译. Python3 程序开发指南[M]. 北京: 人民邮电出版社, 2015
- [10] (加)Dusty Phillips 著, 肖鹏, 常贺, 石琳译. Python3 面向对象编程[M]. 北京: 电子工业出版社, 2015
- [11] (澳) Robert Layton 著, 杜春晓译. Python 数据挖掘入门与实践[M]. 北京: 人民邮电出版社, 2016. 7
- [12] (土)埃塞姆·阿培丁著. 机器学习导论[M]. 北京: 机械工业出版社, 2016.
- [13] 理查德 劳森著. 用 Python 写网络爬虫[M]. 北京: 人民邮电出版社, 2016.
- [14] Ryan Mitchell 著, 陶俊杰, 陈小莉译. Python 网络数据采集[M]. 北京: 人民邮电出版社, 2016.
- [15] 王光宏. 数据挖掘综述[J]. 同济大学学报, 2004, (2).
- [16] 穆瑞辉, 付欢. 浅析数据挖掘概念与技术[J]. 新乡教育学院学报, 2008, 21 (3).

- [17] 杨定中, 赵刚, 王泰. 网络爬虫在 Web 信息搜索与数据挖掘中应用[J]. 计算机工程与设计, 2009, 30(24).
- [18] 姜奇平. 大数据时代到来[J]. 互联网周刊, 2012(2).
- [19] 肖天灿, 陈志刚. 数据挖掘概念与国内外现状[J]. 计算机光盘软件与应用, 2012, (20).
- [20] 马建光, 姜巍. 大数据的概念、特征及其应用[J]. 国防科技, 2013, 32(2).
- [21] 黄翠萍. 数据挖掘概念综述[J]. 数字技术与应用, 2014, (1).
- [22] 于笑笑. 数据挖掘中的决策树分类[J]. 数字技术与应用, 2014, (1).
- [23] 耿丽娟. 用于大数据分类的 KNN 算法研究[J]. 计算机应用研究, 2014, (5).
- [24] 方巍, 郑玉, 徐江. 大数据: 概念、技术及应用研究综述[J]. 南京信息工程大学学报, 2014, 6(5).
- [25] 周中华, 张惠然, 谢江. 基于 Python 的新浪微博数据爬虫[J]. 计算机应用, 2014, 34(11).
- [26] 吴剑兰. 基于 Python 的新浪微博爬虫研究[J]. 无线互联科技, 2015, (6).
- [27] 谢妞妞. 决策树算法综述[J]. 软件导刊, 2015, (11).
- [28] 孙海华. ‘大数据’时代观点综述[J]. 清华大学学报, 2015, (26).
- [29] 方蓉. 基于关联规则的数据挖掘算法的分析及应用[J]. 电子测试, 2016, (1).
- [30] 陈琳, 任芳. 基于 Python 的新浪微博数据爬虫程序设计[J]. 信息系统工程, 2016, (9).
- [31] 邹祎. 数据挖掘技术综述[J]. 信息通信, 2016, (12).
- [32] 熊富琴. Web 数据挖掘综述[J]. 电子世界, 2016, (18).
- [33] 夏火松, 李保国. 基于 Python 的动态网页评价爬虫算法[J]. 软件工程师, 2016, 19(2).
- [34] 柳萌萌, 赵书良, 韩玉辉, 苏东海, 李晓超, 陈敏. 多尺度数据挖掘方法[J]. 软件学报, 2016, 27(12).
- [35] 钱程, 阳小兰, 朱福喜. 基于 Python 的网络爬虫技术[J]. 黑龙江科技信息, 2016, (36).

- [36] 王杰, 蔡良建, 高瑜. 一种基于决策树的多实例学习算法[J]. 郑州大学学报, 2016, 48(1).
- [37] 田欣. 决策树算法的研究综述[J]. 现代营销, 2017, (1).
- [38] 孟强, 李海晨. Web 数据挖掘技术及应用研究[J]. 电脑与信息技术, 2017, (1).
- [39] 章泳. 大数据下的 Web 数据集成与挖掘[J]. 电子技术与软件工程, 2017, (12).
- [40] 李青春. 基于关联规则算法的数据挖掘研究[J]. 软件导刊, 2017, 16(2).
- [41] 毛国君, 胡殿军, 谢松燕. 基于分布式数据流的大数据分类模型和算法[J]. 计算机学报, 2017, 40(1).
- [42] 郑东飞. 基于 XML 的 Web 数据挖掘技术研究与实现[D]. 济南: 山东大学(硕士), 2005.
- [43] 易明. 基于 Web 挖掘的电子商务个性化推荐机理与方法研究[D]. 武汉: 华中科技大学(博士), 2006.
- [44] 王彤. 数据挖掘的新技术研究[D]. 天津: 天津大学(博士), 2006.
- [45] 罗兵. 支持 AJAX 的互联网搜索引擎爬虫设计与实现[D]. 杭州: 浙江大学(硕士), 2007.
- [46] 卢东标. 基于决策树的数据挖掘算法研究与应用[D]. 武汉: 武汉理工大学(硕士), 2008.
- [47] 杨毅超. 基于 Web 数据挖掘的作物商务平台分析与研究[D]. 长沙: 湖南农业大学(硕士), 2008.
- [48] 于宝华. 基于数据挖掘的高考数据分析[D]. 天津: 天津大学(硕士), 2009.
- [49] 张轲智. 基于 web 的数据挖掘系统设计与实现[D]. 成都: 电子科技大学(硕士), 2013.
- [50] 王华. 基于 YARN 的数据挖掘系统的设计与实现[D]. 北京: 北京邮电大学(硕士), 2015.
- [51] 马联帅. 基于 Scrapy 的分布式网络新闻抓取系统设计与实现[D]. 西安: 西安电子科技大学(硕士), 2015.

- [52] 邵臻. 基于特征分析和数据降维的复杂数据预测与分类方法研究[D]. 武汉:武汉理工大学(博士), 2015.
- [53] 邝洪胜. 基于 Python 的电商导购 APP 设计与实现[D]. 广州:华南理工大学(硕士), 2015.
- [54] 罗一纾. 微博爬虫的相关技术研究[D]. 哈尔滨:哈尔滨工业大学(硕士), 2015.
- [55] 尚丹丹. 基于虚拟机的 Hadoop 分布式聚类挖掘方法研究与应用[D]. 哈尔滨:哈尔滨理工大学(硕士), 2015.
- [56] 丰玄霜. 基于数据挖掘的用户上网行为分析[D]. 北京:中央民族大学(硕士), 2016.
- [57] 何江. 基于决策树的教学信息挖掘系统的研究与实现[D]. 长春:吉林大学(硕士), 2016.
- [58] 陈旭. 一种改进的决策树分类算法[D]. 武汉:华中师范大学(硕士), 2016.
- [59] 李海涛. 基于 Hadoop 的决策树算法改进及林业数据分类预测研究[D]. 哈尔滨:东北林业大学(硕士), 2016.
- [60] 邓蓓蓓. 基于信息增益的量化算法及其在决策树中应用的研究[D]. 广州:广东工业大学(硕士), 2016.
- [61] 邱磊. 基于决策树 C4.5 算法剪枝策略的改进研究[D]. 武汉:华中师范大学(硕士), 2016.
- [62] 李冬梅. 朴素贝叶斯与决策树混合分类方法的研究[D]. 大连:大连海事大学(硕士), 2016.
- [63] 陈莹. Web 数据挖掘技术在在线花店推荐中的应用研究[D]. 武汉:湖北工业大学(硕士), 2017.
- [64] 刘哲. 大数据时代下网络招聘有效性研究[D]. 长沙:中南林业科技大学(硕士), 2017.
- [65] 杨余垒. 改进的关联规则算法在慢性病数据挖掘中的研究[D]. 杭州:浙江理工大学(硕士), 2017.
- [66] 安子建. 基于 Scrapy 框架的网络爬虫实现与数据抓取分析[D]. 长春:吉

林大学(硕士), 2017.

[67] 郝文静. 基于图书评论的数据挖掘技术研究[D]. 北京:北方工业大学(硕士), 2017.

[68] 郇益斌. 基于训练集聚类的 KNN 算法及其应用研究[D]. 北京:山东科技大学(硕士), 2017.

[69] 李巨伟. 数据挖掘在高校图书馆个性化推荐服务中的应用[D]. 石家庄:河北科技大学(硕士), 2017.

## 个人简介

刘熠，男，汉族，1991 年 02 月 28 日出生于湖北省荆州市，身体健康。2017 年 6 月通过了全国英语六级考试，具有较强的听、说、读、写能力；完成了培养方案上规定的理论知识学习，修完了 16 门课程及 2 门实践课程，成绩均合格，累计获得 34 个学分，其中学位课 10 门，共 20 个学分，平均成绩 87.2 分；选修课 6 门，共 12 个学分，平均成绩 87.5 分，成绩优秀。

### **主要学习工作经历：**

2013 年 06 月毕业于长江大学计算机科学学院，获工学学士学位；

2015 年 09 月考入长江大学计算机科学学院，攻读计算机专业硕士学位，将于 2018 年 06 月毕业并获工学硕士学位。

### **在校期间参加的科研项目及获奖情况（名称、等级、授奖单位及时间）：**

2015 年 8 月—2017 年 4 月参与钻井信息统计分析软件项目；

### **在校期间发表的主要学术论文（论文题目、刊物名称、时间）：**

2017 年 02 月以第一作者身份在《中国新通信》（半月刊第 4 期）上发表了题为“云计算概念及核心技术综述”的论文 1 篇；

经过一年半的研究完成了题为“**基于 python 的 web 数据挖掘技术的研究与实现**”硕士学位论文。

## 研究生学位论文原创性声明和版权使用授权书

### 原创性声明

我以诚信声明：本人呈交的硕士学位论文是在梁少华副教授指导下开展研究工作所取得的研究成果。文中关于“基于 python 的 Web 数据挖掘技术研究”的结论系本人独立研究得出，不包含他人研究成果。所引用他人之思路、方法、观点、认识均已在参考文献中明确标注，所引用他人之数据、图件、资料均已征得所有者同意，并且也有明确标注，对论文的完成提供过帮助的有关人员也已在文中说明并致以谢意。

学位论文作者（签字）： 刘熠

签字日期： 2018 年 5 月 15 日

### 版权使用授权书

本人呈交的硕士学位论文是本人在长江大学攻读硕士学位期间在导师指导下完成的硕士学位论文，本论文的研究成果归长江大学所有。本人完全了解长江大学关于收集、保存、使用学位论文的规定，即学校有权保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权长江大学可以将本学位论文的全部或部分内容编入有关数据库，可以采用影印、缩印、数字化或其它复制手段保存论文，学校也可以公布论文的全部或部分内容。（保密论文在解密后遵守本授权书）

学位论文作者（签字）： 刘熠

签字日期： 2018 年 5 月 15 日