基于深度学习的
招聘与求职双向推荐模型

[2023 年 4 月 28 日]

摘要

近年来,python 作为一门编程语言,提供了高速的高级数据结构,以及简 单有效地面向对象编程，随者人工智能和大数据的发展，python 逐渐成为流行 的编程语言；在新时代背景下，大学生毕业人数不断增加，大学生求职问题已成为广泛关注的社会热点。而且受疫情影响，诸多企业的招聘都改为线上进行，脱离时间和空间的限制，招聘需求不断上涨，从而出现就业竞争压力大、招聘与求职信息不对称等现象,双向职业推荐模型连接了企业与求职者的需求,利用 python 语言对竞双向职业推荐模型的建立具有重大意义。

通过从泰迪内推平台（https://www.5iai.com/#/index）的“找工作”页面和“找人才”页面,爬取所有招聘与求职信息并整理并形成文件:result1-1.csv 以及 result1-2.csv，数据的爬取主要采用了 python 的爬虫技术。

为了能准确的形成双向推荐模型我们们结合了 BI-LSTM 算法、文本匹配、Attention-LSTM 模型进行实现，基于 python 爬取得信息数据，我们对数据得空缺值处理以及异常值进行了处理，最后我们使用里 python 的 matplotlib 库、seaborn 库、对数据信息进行多个维度的用户画像，对数据之间的关系进行了数据可视化分析。基于预处理后的数据，构建岗位匹配度和求职者满意度的模型，使用 BI-LSTM 算法、文本匹配等算法，计算两表中的岗位匹配度以及满意度，每条招聘信息提供岗位匹配度非 0 的求职以及每位求职者提供求职者满意度非 0 的招聘信息存放在指定的表格中。

目录

1.绪论

1.1 研究背景及意义

在新时代背景下，随着大学生毕业人数不断增加，大学生求职问题已成为广泛关注的社会热点。而且受疫情等诸多因素的影响，诸多企业的招聘都改为线上进行，脱离时间和空间的限制，招聘需求不断上涨，有近六成企业招聘需求增加，其中需求量较大的科技研发、数字化、蓝领技能岗位都存在不同程度的人才短缺。但从人才供给来看，应届生数量增加，2022 年高校毕业生达到创纪录的 1076 万人，而且部分企业校招开展暂缓或推迟，因此出现校招需求缩减或冻结，这些因素都加剧了应届生就业的严峻形势。基于种种因素，出现就业竞争压力大、 招聘与求职信息不对称等现象。

用户发布的求职信息内容蕴含了丰富的职业信息，这些内如给分析构建多个方面去建立求职者画像，为就业者以及企业提供便携的双向推荐。

求职发展是大势所趋，充分利用求职者和企业招聘信息构建双向推荐模型。从数据中提取相关的招聘要求，与企业招聘人员要求相匹配。企业可以利用本结果侧重了解求职者的技能以及态度，求职者在选择企业时也将更具有目的性，充分利用线上给予的资源。

1.2 章节安排

根据前面相关内容的分析，我们针对用户企业双向推荐模型的研究思路主要从几节相应介绍：

1） 第 3 主要介绍我们对泰迪内推网站的数据爬取过程

2） 第 4 主要是介绍我们第二题的求职者以及企业多个维度的用户画像，使用 matplotlib 库、seaborn 库，进行图像绘画。

3） 第 5 主要介绍我们第三题的构建岗位匹配度和求职者满意度的模型，使用 BI-LSTM 算法、文本匹配、Attention-LSTM 模型等。

2.相关技术

2.1 BI-LSTM 算法

前向的 LSTM 与后向的 LSTM 结合成 BiLSTM[1]，结构如下图 1 所示。前向 LSTM 可以获取输入序列的过去数据信息，后向 LSTM 可以获取输入序列的过去数据信息，对时间序列实现向前和向后两次 LSTM 训练，进一步提高特征提取的全局性和完整性。

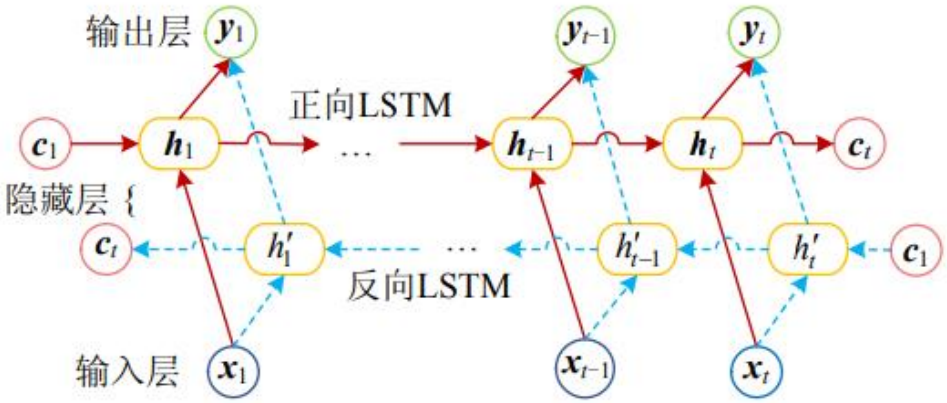


图 2-1 Bi-LSTM 网络

在 t 时刻下，Bi-LSTM 的隐藏层输出值 H_t 由正向 h_t 和反向 h_t 组成：

$$\vec{h}_t = \overrightarrow{\mathrm{LSTM}}(h_{t-1}, x_t, c_{t-1}), t \in [1, T]$$

$$\overleftarrow{h}_t = \overleftarrow{\mathrm{LSTM}}(h_{t+1}, x_t, c_{t+1}), t \in [T, 1]$$

$$H_t = [\vec{h}_t, \overleftarrow{h}_t]$$

LSTM 的全称是 Long Short-Term Memory,它是 RNN 的一种。LSTM 模型是由 t 时刻的输入词 Xt，细胞状态 Ct，临时细胞状态 Ct，隐层状态 ht，遗忘门 ft ，记忆门 it，输出门 ot 组成。LSTM 的计算过程可以概括为，通过对细胞状态中信息遗忘和记忆新的信息使得对后续时刻计算有用的信息得以传递，而无用的信息被丢弃，并在每个时间步都会输出隐层状态 ht，其中遗忘、记忆与输出由通过上个时刻的隐层状态 ht-1 和 X t 当前输入计算出来的遗忘门 ft，记忆门 it，输出门 ot 来控制。总体框架如下图 2-2：

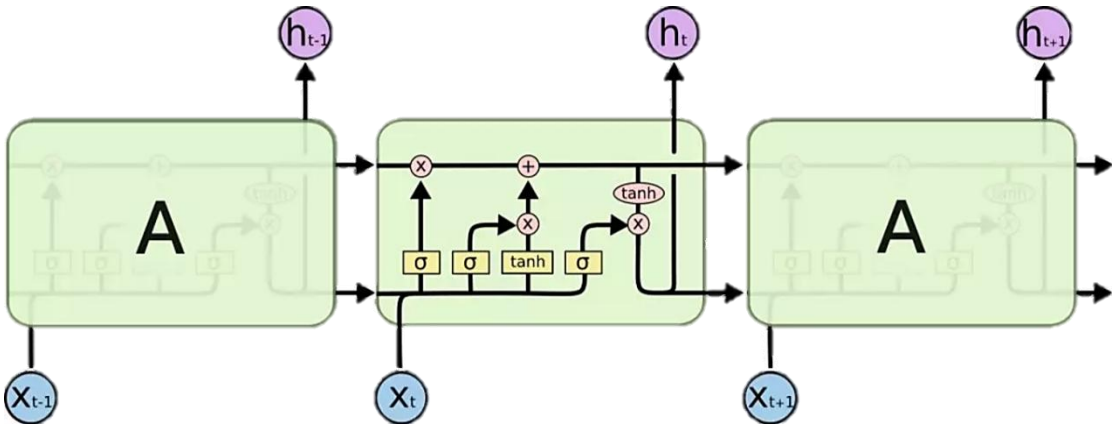


图 2-2 LSTM 网络

2.2 文本匹配

文本匹配旨在从两端文本中挖掘内在的语义特诊，预测文本间相关行或者矛盾性。文本匹配任务是自然语言处理中重要的研究方向，无论是在信息检索、问题回答还是复述识别等任务中都扮演着重要角色。传统的文本匹配方法依赖于预定义的模板和人工提取的规则。随着深度学习的发展，深度神经网络已经普遍应用于自然语言处理任务中，以节省人工提取特征所耗费的成本和时间。文本匹配任务旨在给定两段文本 Q 和 D，通过提取文本中存在的语义信息和相似度特征来给出两段文本的相似度值，由最终的相似度值可以得知两段文本的内容是否属于相似的描述。基于深度学习的文本匹配模型大致可以分为两类：基于表示的文本匹配模型和基于交互的文本匹配模型。基于表示的文本匹配模型通过对文本进行预处理以及构建索引，能够更好地提取每段文本里面的信息，但是在信息表示的过程中容易失去语义焦点，难以衡量词在上下文中的重要性。基于交互的文本匹配模型虽然可以较好地把握语义焦点，针对上下文进行更好的建模，但是却忽视了全局信息，会造成无法刻画出全局匹配信息的后果[2]

2.3 Matplotlib 库

Python 开发中是专门用于开发 2D 图表(包括 3D 图表) 以渐进、交互式方式实现数据可视化，它以各种硬拷贝格式和跨平台的交互式环境生成出版质量级别的图形，可视化是在整个数据挖掘的关键辅助工具，可以清晰的理解数据，从而调整我们的分析方法。 能将数据进行可视化,更直观的呈现 使数据更加客观、

更具说服力。

2.4 Attention-LSTM 模型

将 LSTM 层的输出向量作为 Attention 层的输入。注意力机制的本质为计算某一特征向量的加权求和。 本文采用的是乘法注意力机制中的 Scaled Dot-Product Attention 方法，其计算主要分为 3 个步骤：

1. 将 query 和每个 key 进行点积计算得到权重
2. 使用 Softmax 函数对权重归一化处理。
3. 将权重和对应的 value 加权求和获得 Attention 。

综上所述，Attention 层输出的计算公式如下

$$Attention(Q,K,V) = Softmax\left(\frac{Qk^T}{\sqrt{d}}\right)V$$

2.5 tensorflow 库：

TensorFlow 是一个基于数据流编程（dataflow programming）的符号数学系统，被广泛应用于各类机器学习（machine learning）算法的编程实现，其前身是谷歌的神经网络算法库 DistBelief。Tensorflow 拥有多层级结构，可部署于各类服务器、PC 终端和网页并支持 GPU 和 TPU 高性能数值计算，被广泛应用于谷歌内部的产品开发和各领域的科学研究 。

3.招聘信息数据源的确定

本次招聘信息数据源选择泰迪提供的内推平台，采集内容为多家企业发布的招聘信息以及求职者求职信息，共爬取招聘类信息 1574 条，求职类信息 10908 条。本次采集时间为 2023 年 3 月 14 日至 2023 年 3 月 15 日。

数据类型	数据源	采集方法
招聘信息	泰迪内推平台“找工作”页面	Python 爬虫
求职信息	泰迪内推平台“找人才”页面	Python 爬虫

3.1 网络爬虫的基本流程

网页爬虫是通过模拟浏览器向服务器发送请求，从而获取响应结果。网络爬虫的一般过程是对目标页面进行分析，然后爬取信息，最后解析保存数据，如下图所示。

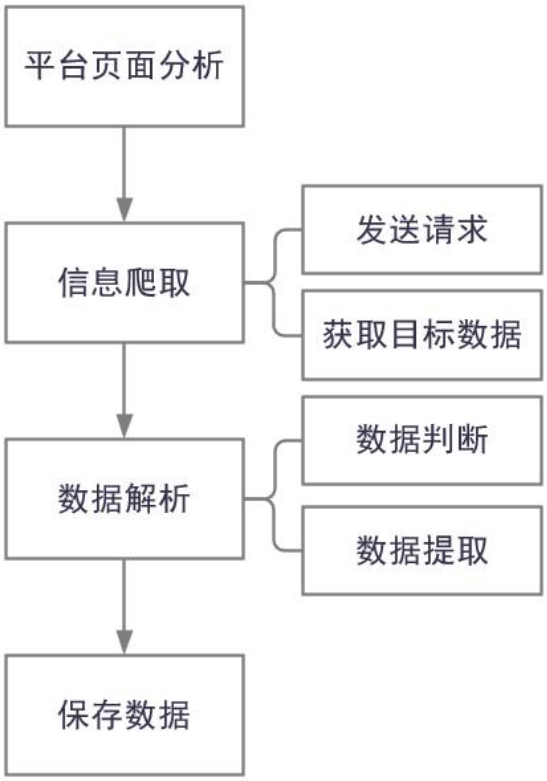


图 3-1 平台页面爬虫基本流程图

平台页面分析主要分为页面结构分析和内容分析两部分。页面结构分析主要包括以下方面：爬取目标页面是静态还是动态的；确定请求方式。内容分析是确定需要采集的内容在页面的什么位置，主要有两种情况：一是待采集的内容就在当前页面，可以直接爬取；二是待采集的内容需要提前制定采集规则[3]，利用当前页面进行跳转获取。信息爬取分为两步，发送请求和获取响应目标内容。利用 Python 模拟浏览器对网页发出请求时主要使用 requests 库，请求方式为 get。返回请求成功时通过 json 形式获取响应的文本数据。数据解析是对获取到的数据进行内容判断和字段提取，选取具有分析价值的字段，由于平台数据格式较为规整，最后将解析后的数据保存到 csv 文件中，分别为 result1-1.csv 和 result1-2.csv。

3.2 数据类招聘信息的采集

3.2.1 平台页面分析

本文使用爬虫程序的目的是采集内推网站的岗位招聘信息和求职信息。在编写爬虫程序之前需要根据采集内容确定网页位置，并从页面结构和内容两方面对其进行分析。

（1）网页结构分析

网页结构分为以下部分：爬取目标页面是静态还是动态的，确定请求方式，是否存在反爬措施。判断爬取目标页面是静态还是动态的可以通过爬取部分头部信息，根据爬取的 body 和 JavaScript 跳转得出需要使用动态爬虫；通过 F12 观察 network 请求列表得到爬取两大网站的规律：pageNumber 表示浏览的页面为分页模块的第几页面，PageSize 表示当前浏览页面的数据量，点击请求链接和响应标头部分发现该链接的请求方式是 get。

（2）内容分析

泰迪内推“找工作”页面筛选条件主要有职位和企业两部分，通过观察职位页面的招聘数据得出每一条招聘信息包含工作岗位名称、薪资范围、学历要求、工作经验、招收人数、公司名称、公司类型、公司规模、发布时间和职位关键词等 10 类信息。职位页面每一页最多显示 10 条招聘信息，共 158 页数据，累计 1574 条招聘信息。企业页面主要是企业的基本信息，包含企业名称、职位关键词、企业规模和招收人数 4 类信息，每一页最多显示 16 家企业信息，共 36 页数据，累计 568 条招聘信息。本次需要采集的数据为内推网站的招聘信息，通过对比分析发现求职页面信息包含企业页面信息，经过初步分析后决定爬取求职页面中的招聘信息。

图 3-2 泰迪内推平台页面

3.2.2 信息爬取

通过对内推网站的页面结构和内容分析，将进一步爬取目标信息，使用 requests 库的 get 方法向浏览器发出请求，其使用 requests 方法的语法格式如下：response=request.get(‘目标网址’)。请求成功后获取响应的文本数据，将请求到的数据存储为 json 格式文件，爬取岗位详情页面过程中发现第 91 个页面后出现大量的完全空页，使用 Python 代码观察 min(os.path.getsize(path)) 发现完全空页文件大小在 44-48 字节之中，直接对文件大小进行判断：文件大于 50 字节且拥有关键的目标值才对其进行处理，否则均给空值。

3.2.3 数据解析

浏览 json 文件下的 data 目标数据发现有我们需要进一步分析的数据"工作 id"，"工作更新时间"，"工作岗位名称"，"工作岗位最低工资"，"工作岗位最高工资"，"工作岗位要求经验"，"工作岗位要求学历"，"工作岗位招收人数"，"工作岗位具体地址","公司名称"，"公司所属类型"，"公司成立模式"，"公司规模"等，由于大部分数据在 content 目录下，直接通过列表索引或切片索引进行目标数据的获取。

将获取到的所有岗位数据根据 id 划分到每一个列表子目录下，依据 id 排序

划分到总列表，将其分别保存至 csv 文件中，文件分别为“result1-1.csv”和“result1-2.csv”。招聘信息保留招聘信息 ID、公司名称、招聘岗位、工作岗位最低工资、工作岗位最高工资、工作岗位要求经验、工作岗位要求学历、工作岗位招收人数、工作岗位具体地址、公司所属类型、公司成立模式、公司规模和岗位技能等 13 个字段特征；求职信息保留求职者 ID、姓名、性别、应聘工作岗位、工作经验、最低薪资期望、最高薪资期望、所在城市、上传时间和个人技能等 10 个字段特征来做进一步分析。

文件名	字段特征
result1-1.csv	招聘信息 ID、公司名称、招聘岗位、工作岗位最低工资、工作岗位最高工资、工作岗位要求经验、工作岗位要求学历、工作岗位招收人数、工作岗位具体地址、公司所属类型、公司成立模式、公司规模、岗位技能
result1-2.csv	求职者 ID、姓名、性别、应聘工作岗位、工作经验、最低薪资期望、最高薪资期望、所在城市、上传时间、个人技能

4.招聘与求职信息分析

4.1 数据预处理

从泰迪内推平台的“找工作”页面和“找人才”页面，爬取的数据，较为规整，但为了能让计算机简单有效地处理文本信息，需要对原始的数据变得有规则、有结构、有组织的数据，需要进行数据预处理。

图 4-1 招聘信息页面

泰迪内推网站中“找工作”页面中，每条职位招聘信息排版样式如上图所示，顶部为岗位名称、薪资范围、学历要求、岗位要求、招聘人数、岗位发布时间、投递截止时间等基本信息；左下侧为职位关键词、技能要求、职位描述等具体信息，右侧为公司的基本信息，包括所属行业、公司名称、公司类型等。

图 4-2 求职信息页面

泰迪内推网站中“找人才”页面中，每条求职信息排版样式如上图所示，顶

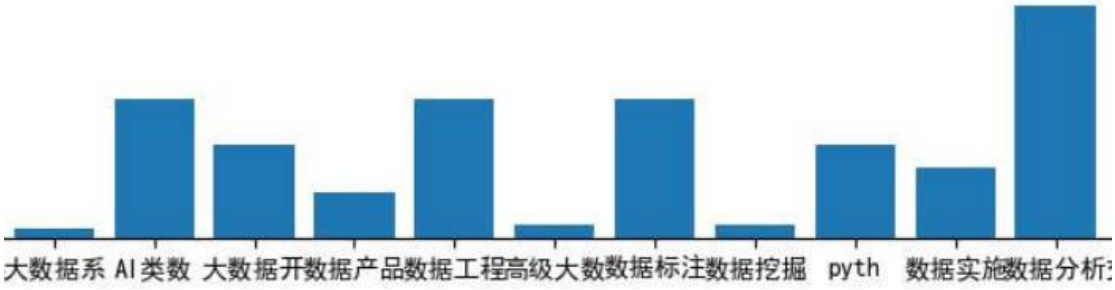
部为期望岗位名称、期望薪资；左侧求职者的基本信息，包括姓名、性别、年龄、工作经验、政治面貌、求职意向、简历关键词、工作经历、项目经历、竞赛经历等具体求职信息。

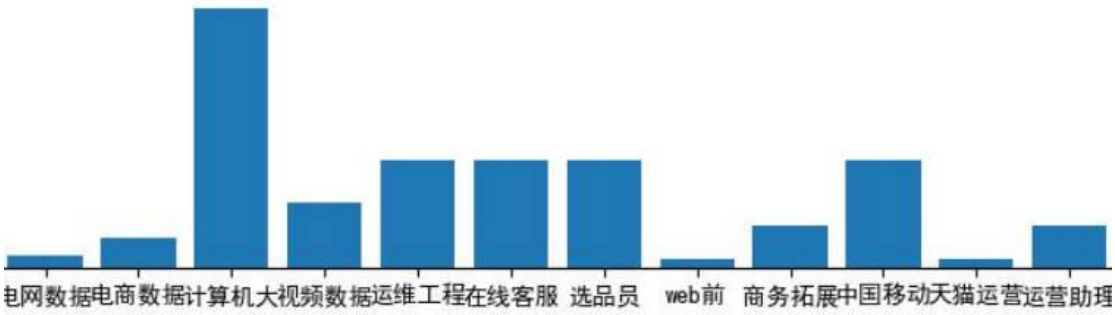
4.1.1 数据清洗

对所得的数据进行缺失值的处理，减少缺失值对画像的不准确性，在分析求职者以及企业招聘数据时，使用 python 的 isnull 对数据的每一列进行计算缺失值，相关数据中并没有出现缺失值，如果相关数据大部分为空值，删去相关的空值数据。

4.2 企业招聘信息

4.2.1 招聘岗位分析





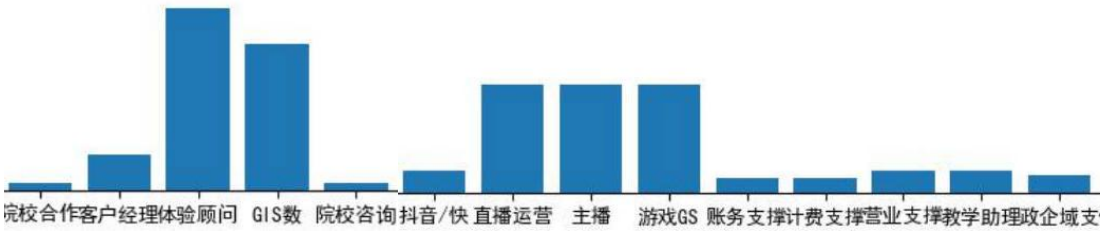


图 4-3 工作岗位柱状图

由于招聘信息中企业发布的工作岗位数据类型较多，本文截取主要的职位进行展示，通过统计的数据可以看出在“大数据+”和“人工智能”领域中，对“数据分析”、“计算机”、“数据产品”、“GIS”类的岗位需求较大，“直播”、“游戏”、“客服”等非计算机类型岗位也有一定需求，求职者可以在该平台上获取适合自身的岗位信息。

4.2.2 学历要求分析

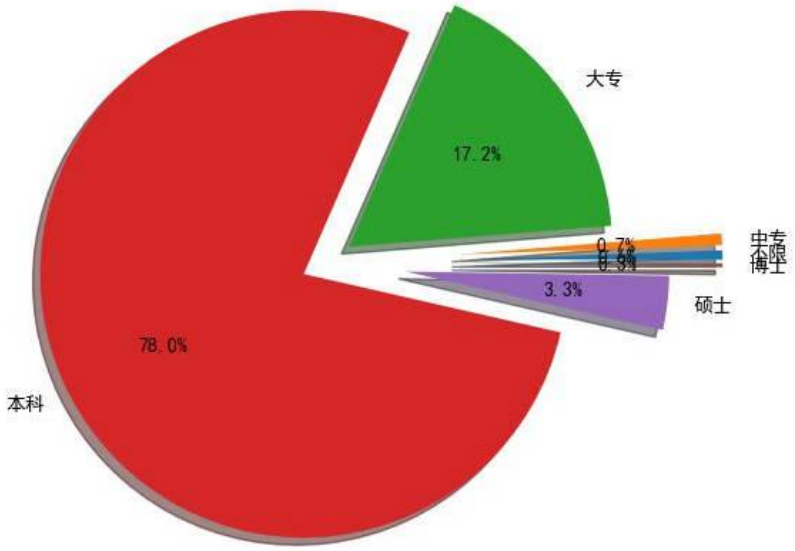


图 4-4 学历要求比例图

根据爬取的招聘信息统计结果可知，招聘信息学历要求最多的是本科，高达78.0%，17.2%的招聘信息要求学历为大专，发布的招聘信息中对学历没有要求的占比为 0.6%，相反，要求硕士学历占比 3.3%，硕士，博士的比例都低于 10%，因此可以说明大多数企业招聘对学历要求是不高的；由上图可以看出，本科和大专的占比是比较大的，企业多为小型或中型企业，从企业招聘现实状况来看，学历要求更多的是本科和大专，博士和硕士相对而言需求较少。

4.2.3 公司成立模式分析

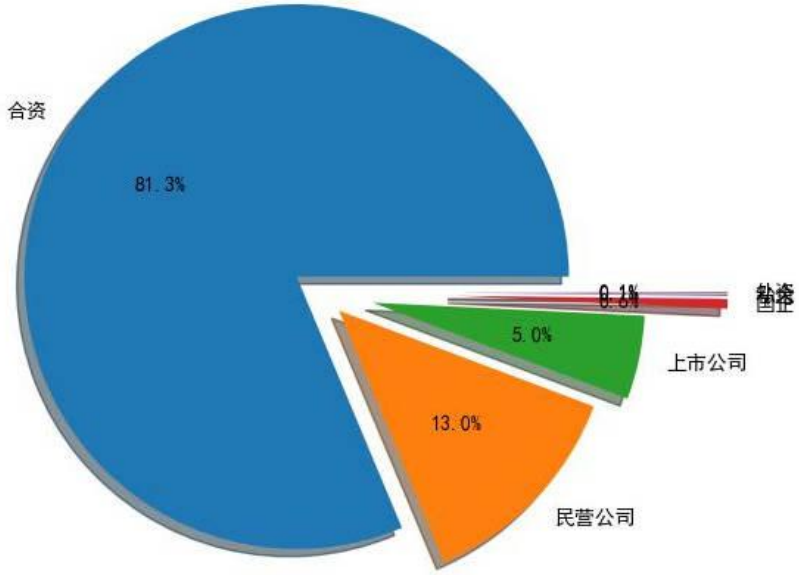


图 4-5 公司成立模式占比图

本文采集的数据中公司成立模式类别为“合资、民营公司、上市公司、国企、私企、外资”五大类，其中公司成立模式为“合资”的比重最大，高达 81.3%，说明目前大多数公司之间相互协作，将有限的资源进行整合，公司长期发展资源得到有效补充。民营公司占比 13.0%，上市公司占比 5.0%，国企和外资占比都低于 3%，反映了在“大数据”和“人工智能”领域的公司多为合资类型。

4.2.4 公司规模统计

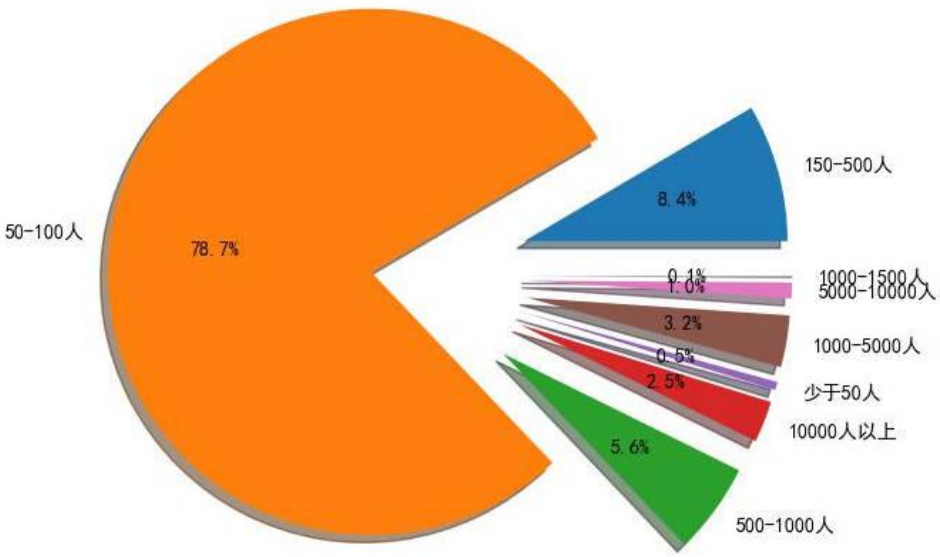


图 4-6 公司规模人数

对获取到的数据 统计 xx 发布数据类岗位企业的规模占比，结果如图 所示。从图可以发现，规模为 50—100 人的企业最多，占比 78.7%，其次是规模为 150—500 人的企业，占比为 8.4%。规模为 1000—1500 人的企业占比最少，占比 0.1%。说明当前的企业规模主要分布在 50—500 人之间，企业大多数处于发展阶段，只有较少部分企业规模较大。

4.2.5 薪资待遇与工作经验、学历要求

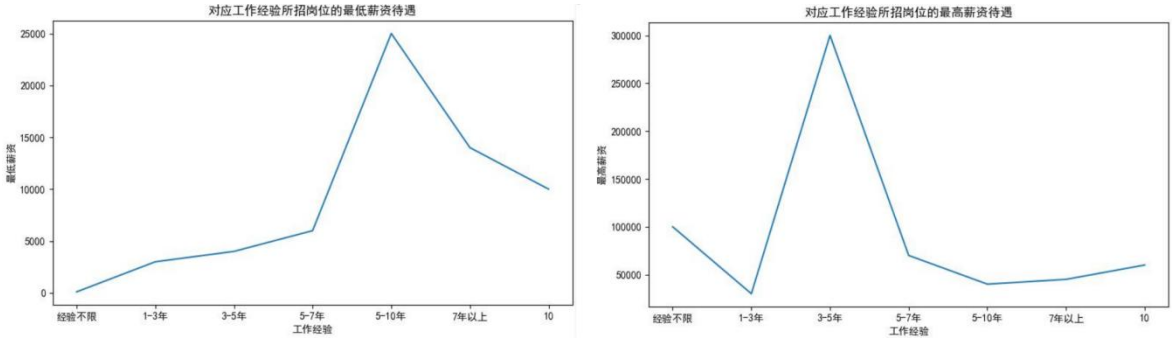


图 4-7 不同工作经验薪资待遇图

根据泰迪内推平台上的招聘信息将工作经验要求划分为以下几种类型：经验不限、1-3 年、3-5 年、5-7 年、5-19 年、7 年以上、10 年。从图中可以看出 5-10 年的工作经验对薪资的提升有很大优势，无论是最低薪资待遇还是最高薪资期望待遇，薪资随着经验年限的增加而不断增长，这可能跟互联网 IT 行业发展时间阶段有关，由于互联网行业近十年来发展较快，技术和专业知识不断更新，所以对该行业方面的经验需求还是比较大的。

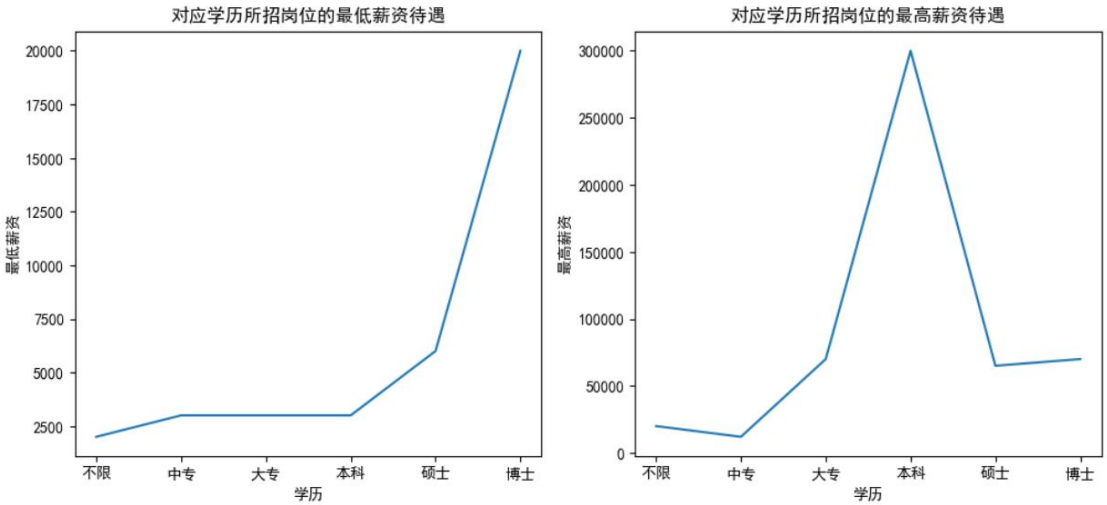


图 4-8 不同学历薪资待遇图

除了上述工作经验会影响薪资，学历要求也是薪资待遇要关注的部分，从图中可以看出在最低薪资待遇里，越高学历的求职者在薪资方面有很大程度的晋升空间，这说明大多数企业对求职者的学历还是比较看重，在高薪资待遇里并非越高学历者薪资越高，据观测，在高薪资待遇里，求职者是已经适应过一段时间的工作培训和工作任务，更多看重的是工作能力和技术而并非学历。

4.2.6 岗位技能分析

对招聘信息中的“岗位技能”做进一步分析统计，细分不同工作岗位名称并加以统计。筛选排除部分没有技能要求的岗位后总共得出有 232 个岗位对技能有要求，标记按工作岗位名称不同计数进行标记，颜色深浅作为工作岗位需要该技能的计数体现，颜色越深代表该技能在岗位的重要性，得出的岗位技能热力图如下图所示。

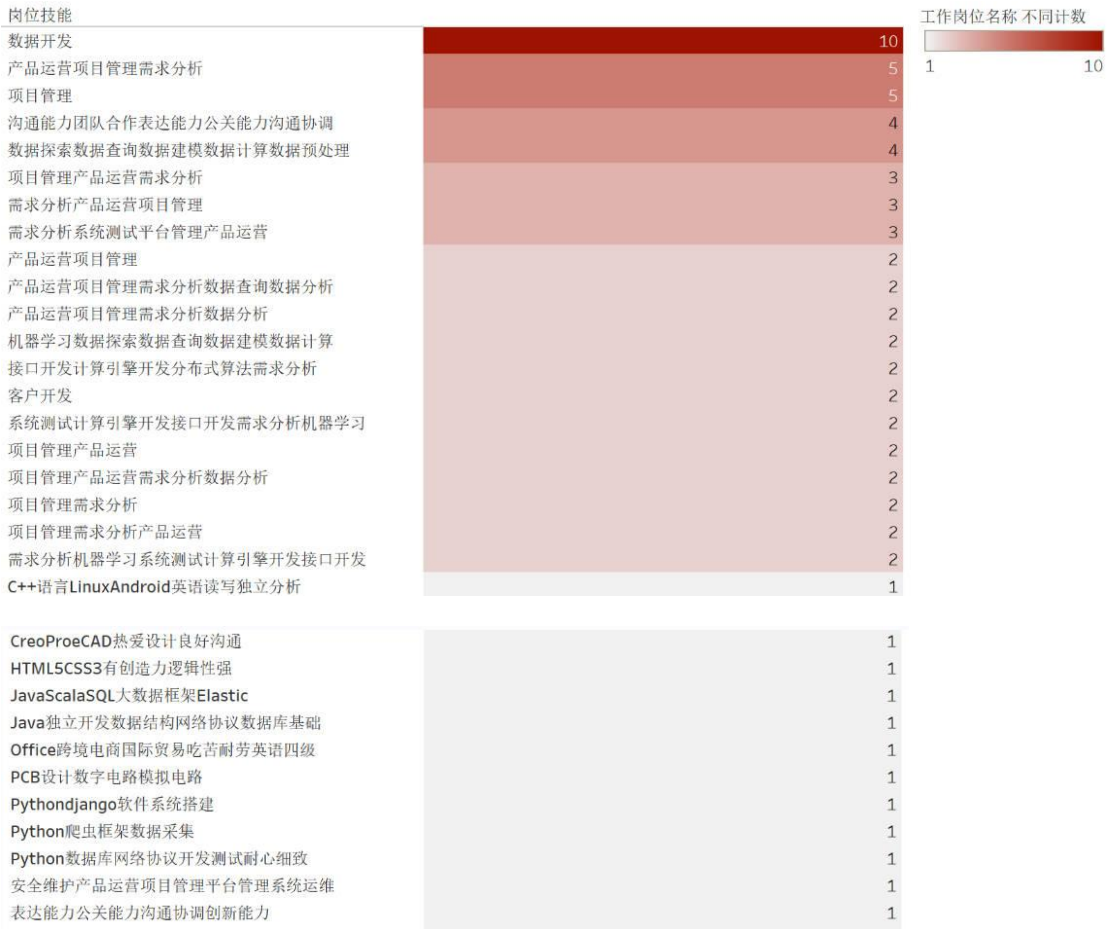


图 4-9 工作岗位技能-招聘岗位热力图

由上图可以看出，大多数公司工作岗位对岗位技能要求较多的是“数据开发”能力，其中“项目管理需求分析”能力和“项目管理”能力也占据重要地位，可以说明在数据开发能力和分析能力是当前“大数据”和“人工智能”领域不断向前发展的重要动力。为进一步筛选岗位技能要求，本文以“广东省”为例，对招聘信息中的工作岗位具体地址、岗位技能等字段分别进行统计分析，筛选出所有的大数据职位与其对应 id，按照 id 将职位描述表中相应的大数据职位的岗位描述和任职要求提取出来，然后利用 jieba 对这些文本进行分词，由于文本中有大量的专业术语如：“数据分析”“数据挖掘”“云计算”等，需要添加自定义的用户词典，将这些专业术语添加进去并词频统计绘制出对应数据的词云图，通过颜色对不同词汇作出区分，根据字体大小强调了词汇的出现频率，字体越大的词汇在岗位技能要求中被提及的次数最多，同时将词云图设计成省份的形状，暗示了企业需要岗位技能要求分布的区域。

图 4-10 广东省岗位技能要求词云图

根据图可以看出。“数据”、“数据分析”、“可视化”、“预处理”、“数据挖掘”、“探索”等词语出现频数较大，说明广东省中大多数企业招聘需要的人才需具有良好的数据分析能力，另外，“计算机”、“神经网络”、“测试”等词语也有一定程度的展现。说明招聘岗位对专业知识也有一定要求。

4.2.7 招聘信息画像

图 4-11 招聘信息画像

通过上述的统计分析描述，招聘信息画像由招聘信息中多维度数据结合而成，主要的画像数据包含公司规模、公司成立模式、企业招聘工作经验要求、学历要求、学历对应岗位的最低薪资和最高薪资待遇、工作经验要求下最低工资和最高工资待遇以及岗位需求量，求职者可通过招聘信息画像匹配符合自身的岗位。

4.3 求职者求职信息

4.3.1 预期岗位

根据爬取到的求职者求职信息进行统计分析，求职者求职信息包含“求职者ID”、“姓名”、“应聘工作岗位”、“性别”、“工作经验”、“最低薪资期望”、“最高薪资期望”、“所在城市”、“上传时间”、“个人技能”等10项数据。下表为截取的部分数据。

求职者ID	姓名	应聘工作岗位	性别	工作经验	最低薪资期望	最高薪资期望	所在城市	上传时间	个人技能
163561035761778	欧先生	数据分析师,数据挖掘工程师,算法工程师	0.0	无经验	3000	4000	广东省,广州市,天河区	2023-03-14	None
163401909257279	赖女士	数据分析师	1.0	无经验	4000	5000	江苏省,南京市,江宁区	2023-03-10	None
163294781611455	梁先生	数据分析师,数据挖掘工程师,机器学习工程师	0.0	1年工作经验	6000	7000	广东省,广州市,天河区	2023-03-07	None
163228658690333	钟先生	数据分析师,数据挖掘工程师,机器学习工程师	0.0	无经验	4000	6000	广东省,广州市,黄埔区	2023-03-05	None
157983380976186	李先生	数据分析师,数据挖掘工程师	0.0	无经验	4000	8000	广东省,深圳市,南山区	2023-03-03	None

表 求职者信息表

岗位名称	词频	岗位名称	词频
数据分析师	10844	机器学习工程师	22
数据挖掘工程师	10781	自然语言处理工程师	14
Hadoop 大数据开发工程师	59	算法工程师	11
其他	42	图像处理工程师	9

表 1 预期岗位词频

首先对求职信息中的“求职岗位”进行分析，由于统计数据文本中有大量的专业术语如：“数据分析师”“数据挖掘工程师”“算法工程师”等，需要添加自定义词典，将这些专业术语添加进去，然后再进行分词，统计高频词汇，由于不同求职者的求职岗位不同，本次提取出前八位的岗位进行分析，从上表可以看出，“数据分析师”出现频率最高，达到10844次，其次是“数据挖掘工程师”，可以看出在“大数据”和“人工智能”领域，求职者岗位意向更多的是关于数据分析和处理方面内容，最后生成的整体预期岗位词云图如图所示。

图 4-12 整体预期岗位词云图

为精确分析求职者的岗位期望，本文依据求职者求职信息中的“性别”字段，分别进行预期岗位的分析。求职信息中获取到的男性数据为 1190 条，女性数据为 975 条。分别使用 collections 库中的 Counter()函数对男性女性数据进行词汇统计，提取出关键词后使用 wordcloud 库绘制对应期望岗位词云图。从图中可以看出男性期望岗位出现频率较高的是“数据分析师”、“数据挖掘工程师”、“Hadoop 大数据开发工程师”等，字体大小强调了词汇的出现频率，字体越大的词汇表示在期望岗位中提及的次数最多。

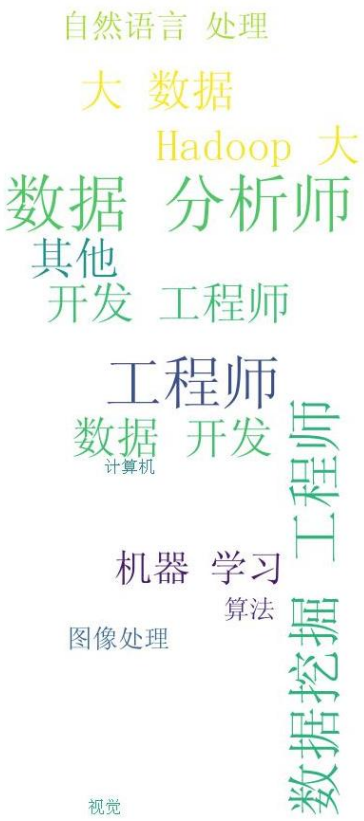


图 4-13 男性期望岗位词云图

图中可以看出女性期望岗位出现频率较高的是“数据分析师”、“数据挖掘工程师”、“Hadoop 大数据开发工程师”等，字体大小强调了词汇的出现频率，字体越大的词汇表示在期望岗位中提及的次数最多。由此可见，在“大数据”、“人工智能”领域内，求职者求职方向和期望岗位大致相同，多为“数据分析师”、“数据挖掘工程师”、“Hadoop 大数据开发工程师”。

图 4-14 女性期望岗位词云图

4.3.2 薪资需求

对求职者求职信息中的薪资进行了统计分析，使用箱盒图直观地识别数据中异常值，能够直观判断数据离散分布情况，了解数据分布状态。

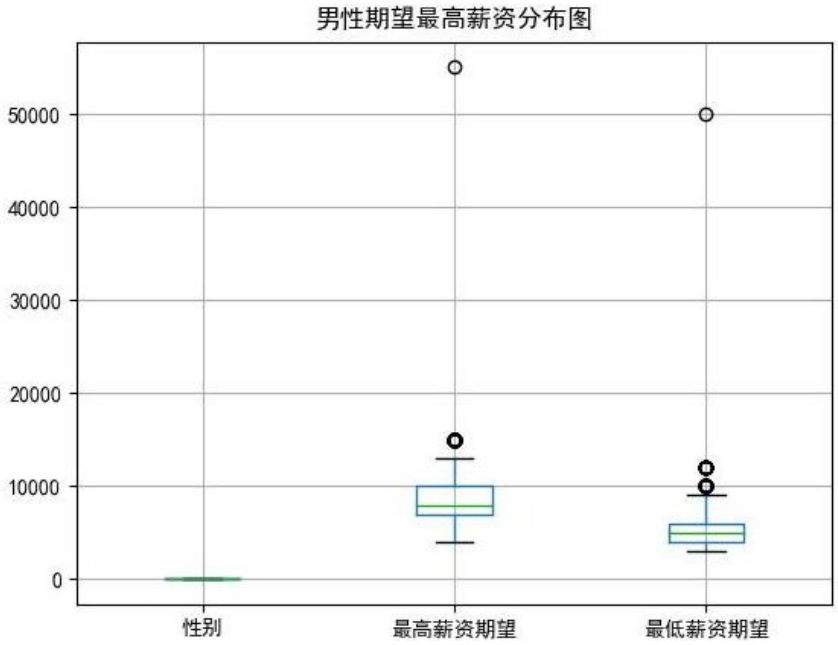


图 4-15 男性期望薪资箱形图

图中可以看出男性薪资期望无论是最高期望还是最低期望，其薪资期望都低于 20000，最高薪资期望比较分散，最低薪资期望分布比较集中，从上下四位数的间距可以看出最低薪资期望分布是平衡的，最高薪资期望则不平衡，最高薪资出现 2 个异常点，最低薪资出现 3 个异常点，但从总体看来最高薪资期望和最低薪资期望两者相差不大，说明男性在薪资期望过程中保持较为平稳的状态。

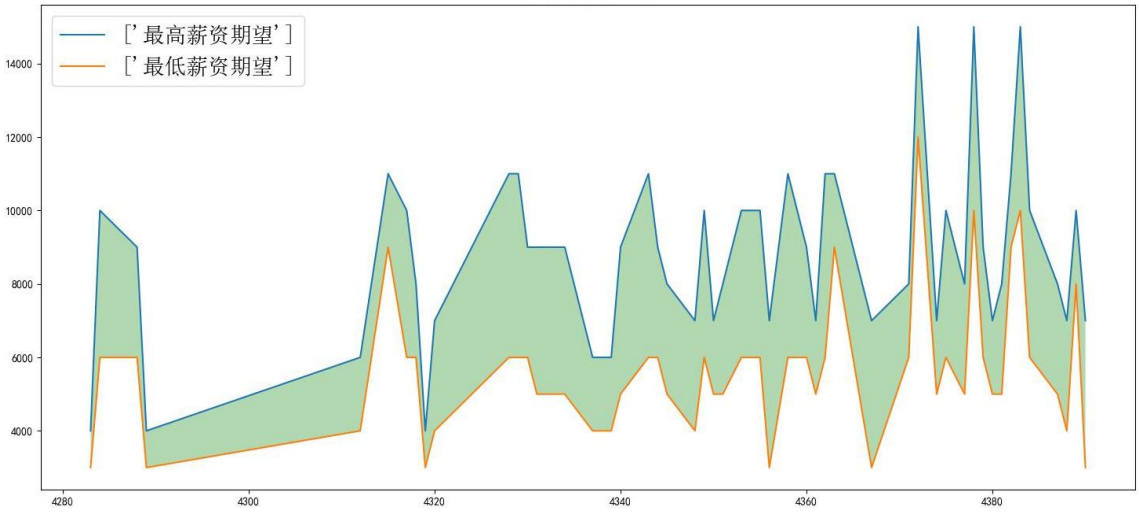


图 4-16 男性薪资幅度图

从男性薪资幅度图可以看出最高薪资期望和最低薪资期望之间有一定的距离幅度，最高薪资和最低薪资变化幅度趋势大致一样，两者在 4285—4310 区域之间变化较为平缓，工资越高则两则变化幅度都比较大。侧面说明工资幅度在到达一定程度后会发生较大改动。

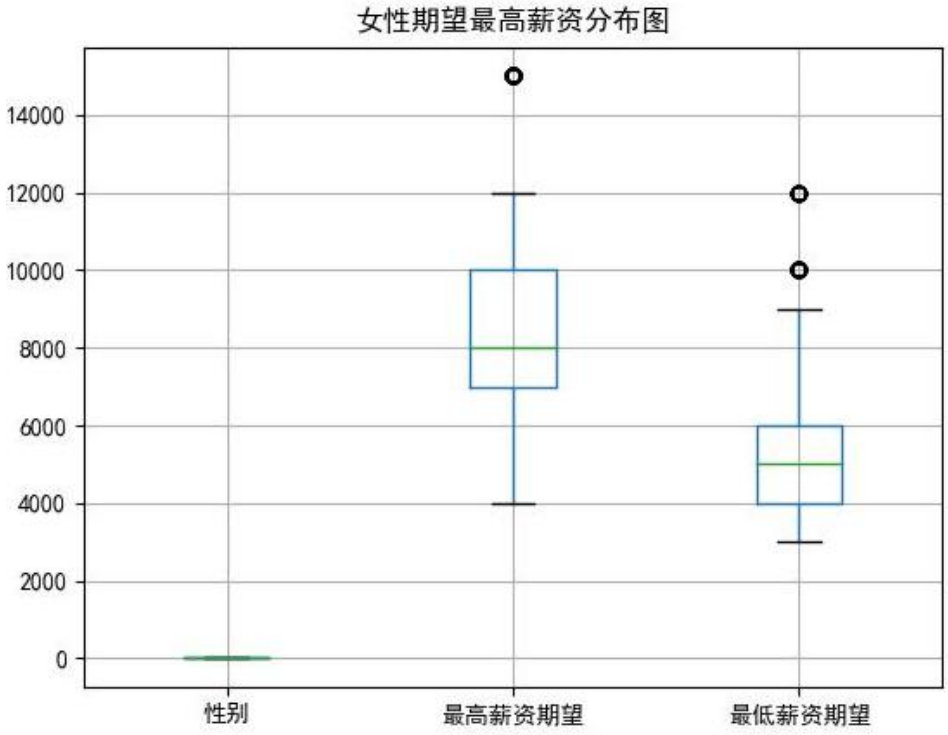


图 4-17 女性期望薪资分布

图中可以看出女性薪资期望无论是最高期望还是最低期望，其薪资期望都低于 16000，最高薪资期望比较分散，最低薪资期望分布比较集中，从上下四位数的间距可以看出最低薪资期望分布是平衡的，最高薪资期望则不平衡，女性最高薪资期望出现 1 个异常点，最低薪资出现 2 个异常点，但从总体看来最高薪资期望和最低薪资期望两者相差不大，说明女性在薪资期望过程中保持较为平稳发展的状态。

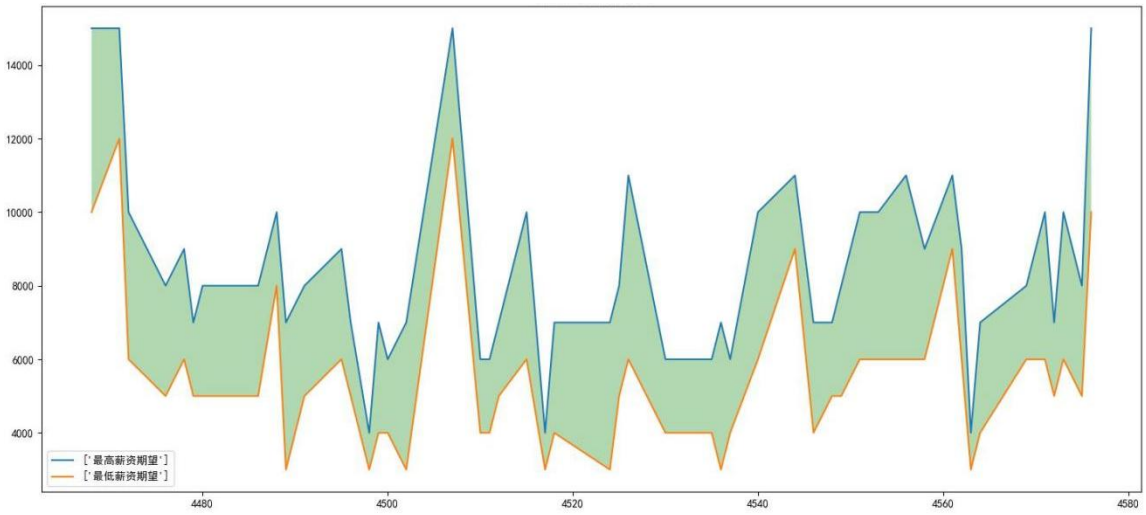


图 4-18 女性薪资幅度图

从女性薪资幅度图可以看出最高薪资期望和最低薪资期望之间有一定的距

离幅度，最高薪资和最低薪资变化幅度趋势大致一样，两者在薪资变化上一直都有较大的波动。侧面说明在薪资上可能会出现较大差异。

4.3.3 知识储备

在进行“个人技能”统计情况中，发现“个人技能”一块出现数据空白，因此需要对空白数据值进行清理，删除空值数据 10279 条，剩余男性数据 180 条，女性数据 184 条。

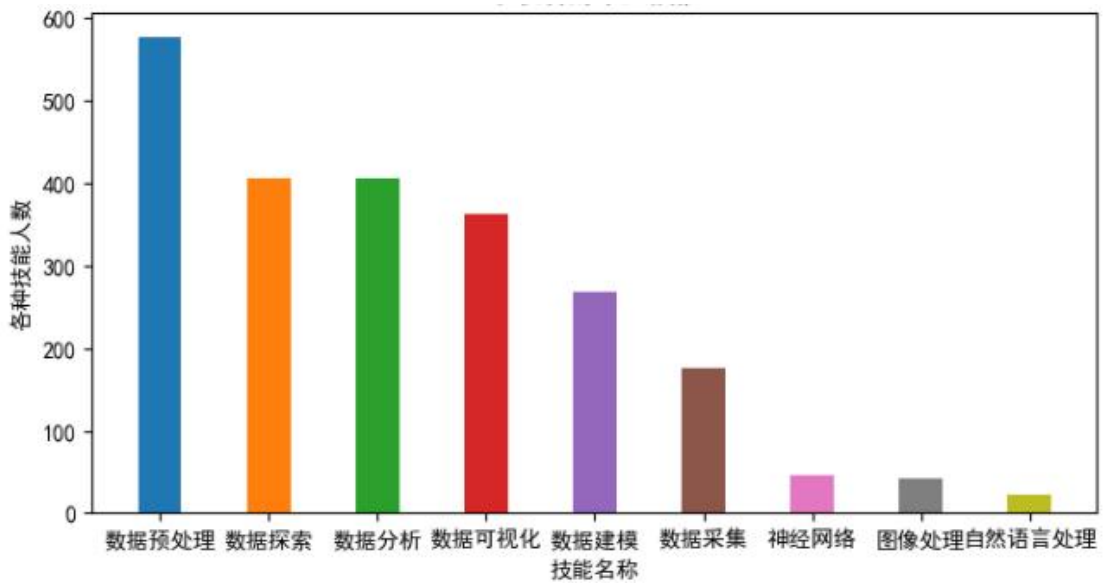


图 4-19 整体个人技能柱状图

根据求职者求职信息的统计结果可知，拥有“数据预处理”能力的人数最多，“数据探索”和“数据分析”能力位列第二第三。依次为“数据可视化”、“数据建模”、“数据采集”、“神经网络”、“图像处理”、“自然语言处理”，下图为个整体人技能词云图。

图 4-20 个人技能词云图

数据处理完成后分别对男性数据和女性数据进行统计，得出男性个人技能词云图和女性个人技能词云图。

图 4-21 男性个人技能词云图

图中可以看出男性总体的个人技能比较突出且较为具备的是“数据可视化”、“数据预处理”、“数据分析”、“建模”、“探索”方面，自然语言、图像处

理、采集、神经网络等内容与“数据”相结合进行形成数据相关的个人能力。

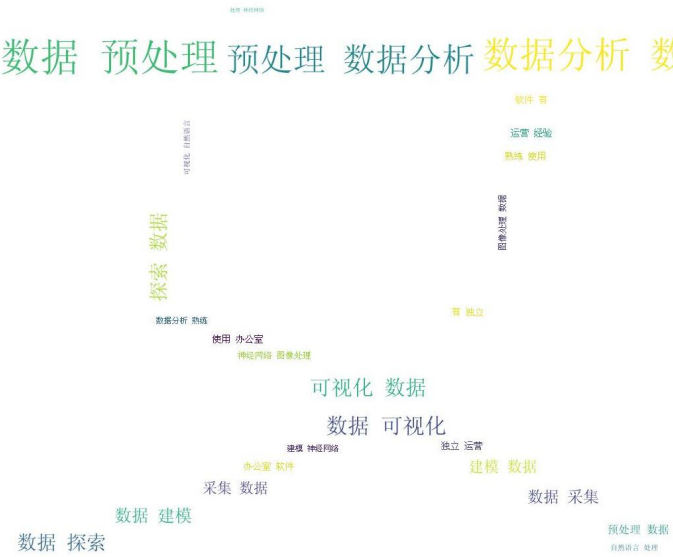


图 4-22 女性个人技能词云图

女性总体的个人技能比较突出且较为具备的是“数据分析”、“数据预处理”、方面，建模、采集、可视化、运营、软件、办公室等内容与“数据”相结合进行形成数据相关的个人能力。

4.3.4 工作经验

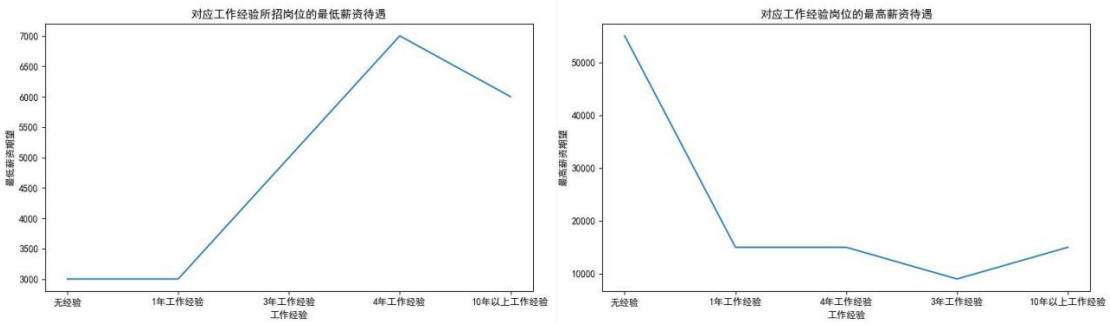


图 4-23 工作经验与薪资待遇情况

对求职者求职信息中的工作经验和薪资待遇进行结合分析得出结果如上图所示，在最低薪资待遇中工资待遇会随工作经验不断上涨，达到 4 年经验临界值时出现下滑迹象；在最高薪资待遇情况中呈相反趋势，无经验者的薪资有望达到最高。

4.2.5 求职者画像

图 4-24 求职者画像

通过上述的统计分析描述，求职者画像由求职者信息中多维度数据结合而成，主要的画像数据包含男性薪资期望、女性薪资期望、个人技能、工作经验对应的最高最低薪资待遇情况求职者可以参考该画像信息找到适合自己的岗位，了解当前其他求职者的能力状况，结合自身找到适合自己将要提升的技能，实现自我增值。

5.构建岗位匹配度和求职满意度的模型

5.1 构建岗位匹配度模型

python 内置函数对数据中的‘工作岗位要求经验’特征的相关文字内容进行了整体的整合和添加标签，‘工作岗位要求经验’的标签大致分为'不限'，'经验不限'，'5-10 年'，'1-3 年'，'3-5 年'，'5-7 年'，'7 年以上'，3，5，1，0,10 等，不利于我们后续的模型训练，从而我们对'工作岗位要求经验'的标签进行重新定义对‘不限’定义为‘经验不限’等，具体标签变更如下图图 5-1 所示：

```
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace(np.nan, '经验不限')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('不限', '经验不限')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('0', '经验不限')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('1', '1年工作经验')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('5-10年', '5年工作经验')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('1-3年', '1年工作经验')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('3-5年', '1年工作经验')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('5-7年', '5年工作经验')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('7年以上', '7年工作经验')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('3', '3年工作经验')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('5', '5年工作经验')
cc['工作岗位要求经验'] = cc['工作岗位要求经验'].replace('10', '10年以上工作经验')

cc['工作经验'] = cc['工作经验'].replace('无经验', '经验不限')
```

图 5-1 标签变更代码

替换了表格中的 1574 条数据的‘工作岗位要求经验’进行替换标签，使得大部分数据符合大多数人的要求，尽可能最大性去选择，无法做到百分百。

我们提取了‘工作岗位名称、工作岗位最低工资、工作岗位要求经验’和‘应聘工作岗位、最低薪资期望、工作经验、个人技能’两者相合并形成一列新的列名命名为‘xdate、ydate’，为后续的岗位匹配度便于计算，将‘求职者 ID、招聘信息 ID、xdate、ydate’形成一个新的表格。部分结果展示如下图图 5-2 所示：

0	1.631113e+18	会计实习生3500.0经验不限	1635610357617786880	数据分析师数据挖掘工程师算法工程师3000经验不限
1	1.631113e+18	技术服务工程师5000.0经验不限	1634019092572798976	数据分析师4000经验不限
2	1.629825e+18	大数据分析师（BI）4500.0经验不限	1632947816114552832	数据分析师数据挖掘工程师机器学习工程师60001年工作经验...
3	1.613440e+18	自然语言处理工程师2000.0经验不限算法研究深度学习	1632286586903330816	数据分析师数据挖掘工程师机器学习工程师4000经验不限...
4	1.613440e+18	爬虫工程师2000.0经验不限Python爬虫框架数据采集	1579833809761861632	数据分析师数据挖掘工程师4000经验不限
...	...	...	...	...
1569	1.463032e+18	前端开发工程师8000.0经验不限数据查询数据计算数据挖掘数据分析数据可视化...	7535876391727459860	数据分析师数据挖掘工程师4000经验不限
1570	1.374181e+18	软件测试工程师6000.01年工作经验系统测试软件系统搭建...	7535876400317394452	数据分析师数据挖掘工程师4000经验不限
1571	1.374177e+18	新媒体运营助理5000.01年工作经验产品运营	7535876404612361748	数据分析师数据挖掘工程师4000经验不限
1572	1.374178e+18	数据产品经理150000.03年工作经验数据探索数据可视化项目管理需求分析	7535876411200290340	数据分析师数据挖掘工程师4000经验不限

图 5-2 部分结果展示

新建列名 xdate，由 result1-1.csv 表格中的“工作岗位名称、工作岗位最低工资、工作岗位要求经验”组成，ydate 由 result1-2.csv 表格中的‘应聘工作岗位、最低薪资期望、工作经验、个人技能’组成。如下图 5-3 所示：

xdate	ydate
[会计, 实习生, 3500.0, 经验, 不, 限]	[数据, 分析师, 数据挖掘, 工程师, 算法, 工程师, 3000, 经验, 不, 限]...
[技术, 服务, 工程师, 5000.0, 经验, 不, 限]	[数据, 分析师, 4000, 经验, 不, 限]
[大, 数据, 分析师, （, BI, ）, 4500.0, 经验, 不, 限]...	[数据, 分析师, 数据挖掘, 工程师, 机器, 学习, 工程师, 60001, 年, 工作...
[自然语言, 处理, 工程师, 2000.0, 经验, 不, 限, 算法, 研究, 深度, ...	[数据, 分析师, 数据挖掘, 工程师, 机器, 学习, 工程师, 4000, 经验, 不,...
[爬虫, 工程师, 2000.0, 经验, 不限, Python, 爬虫, 框架, 数据, ...	[数据, 分析师, 数据挖掘, 工程师, 4000, 经验, 不, 限]...

图 5-3 新建列的数据展示

我们使用 jieba 工具对所得列 xdate、ydate 进行分词，使用自定义的词典，便于我们后续的模型训练，同时新建空列命名为 lable，放置训练结果数值，对所得分词进行词频统计，数据展示图 5-4 词频统计：

词语	次数
经验	3148
数据	2408
工作	2210
年	2208
分析师	1284
不	922
限	922
产品	720
经理	634
工程师	608

表 1-2 词频统计

在处理出使用的数据后，我们随机提取两个岗位信息做分词匹配测试，如使用 sent1 = "高级大数据开发工程师 Hadoop Hive20000.0 经验不限"、sent2 = "项目经理产品经理资质管理员电商大数据研究员 60001 年工作经验"作为测试用例，测试结果显示：两个字符串之间的匹配度为"0.36315152049064636"，即该求职者所具有的技能经验和对岗位的要求工资等信息与该岗位需要具备的内容相关性相关性有但不多。

我们采取了多线程的方式对数据进行模型训练，线程是一个轻量级的子进程，是最小的处理单元；是一个单独的执行路径。线程是进程的子集（部分），一个进程可能由多个线程组成。多线程是一种执行模型，它允许多个线程存在于进程的上下文中，以便它们独立执行但共享其进程资源。总而言之，进程是资源分配的最小单位，线程是 CPU 调度的最小单位。一个线程只能属于一个进程，而一个进程可以有多个线程，但至少有一个线程。

在该列表情况下并不是每一个求职者固定对应到每个岗位，所以将求职者与每一个岗位的关键字进行匹配训练，取平均值作为该岗位面对求职者的匹配度，在内存内核中无法一次性对两百万条数据进行访问、处理和存储，因此使用数据划分、将每一百个求职者，即对应“157,400”条数据做为一个进程去做存储与分析，共计 16 个进程同时并发去访问数据源，达到快速计算与分流计算的目的。线程的示例代码如下图 5-8 线程代码：

```
def onehundred():
    for sent1 in haha['xdate'][0:100]:
        for sent2 in haha['ydate']:
            sent1_ids = sent2index(vocab2id, jieba.lcut(sent1))
            sent2_ids = sent2index(vocab2id, jieba.lcut(sent2))
            sent1_pad = pad_sequences([sent1_ids], maxlen=max_len, padding='post')
            sent2_pad = pad_sequences([sent2_ids], maxlen=max_len, padding='post')
            preds = model.predict([sent1_pad, sent2_pad])
            haha_label_one.append(math.fabs(preds[0]))
    one_list = pd.DataFrame(haha_label_one)
    one_list.to_excel('F:/label.xlsx', sheet_name='onehundred',index=False)
```

图 5-5 线程代码

使用 1574 条岗位数据，10908 个求职者相匹配，是每一个求职者对应每一个岗位数据，选取前 600 个求职者，即“600 * 1574 = 944,400” 大约一百万条数据去做模型匹配度分析，通过 16 个进程获得相应对的匹配度存放 label.xlsx 表格文件中，对得到的匹配度进行整理和查询，我们获得了 944400 条匹配度数据，

我们是在采用了求职者的要求和它求职那个岗位的要求做文本匹配，算出他们每个人的文本匹配的平均匹配率作为最终的匹配模型训练结果，训练结果主要取决于求职者的要求以及岗位要求是主要的文本匹配，同时匹配度越高代表着岗位与求职者会更加合适，在构建匹配岗位匹配度，是以招聘信息与求职者的要求相匹配，构建求职者满意度则反之，岗位匹配度得出的数据如下图 5-6 岗位匹配度：

0	0	1.631110e+18	会计实习生 3500.0经验不限	1.635610e+18	数据分析师数据挖掘工程师算法工程师3000经验不限	0.562923
1	1	1.631110e+18	会计实习生 3500.0经验不限	1.635610e+18	数据分析师数据挖掘工程师算法工程师3000经验不限	0.637812
2	2	1.631110e+18	会计实习生 3500.0经验不限	1.635610e+18	数据分析师数据挖掘工程师算法工程师3000经验不限	0.552606
3	3	1.631110e+18	会计实习生 3500.0经验不限	1.635610e+18	数据分析师数据挖掘工程师算法工程师3000经验不限	0.545203
4	4	1.631110e+18	会计实习生 3500.0经验不限	1.635610e+18	数据分析师数据挖掘工程师算法工程师3000经验不限	0.616072
...	...	...	...	...	...	...
944395	944395	1.482200e+18	大数据产品经理 16000.0经验不限	7.537210e+18	数据分析师数据挖掘工程师8000经验不限	0.295034

图 5-6 岗位匹配度

相同的招聘信息 ID 表示相同的公司不同的岗位，求职者是跟不同的岗位相匹配，得出的匹配数据，为每条招聘信息提供岗位匹配度非 0 的求职者

5.2 求职者满意度模型

求职者满意度模型采用相同的数据清洗，是对工资的处理并不相同，我们采用了 result1-2 表格中‘工作岗位最低工资与工作岗位最高工资’的总和求得平均数’作为平均工资，形成后续作为相匹配的要素，resutl1-1 表提取出求职者 ID、最高薪资期望、工作经验，使其合成一个表格，表格详细内容如下图 5-7 初步表格数据：

	招聘信息 ID	平均工资	工作岗位要求经验	求职者ID	最高薪资期望	工作经验
0	1590.0	12500.0	1-3年	163561035761778	4000	无经验
1	1580.0	8000.0	1-3年	163401909257279	5000	无经验
2	1580.0	16000.0	1-3年	163294781611455	7000	1年工作经验
3	1570.0	11500.0	经验不限	163228658690333	6000	无经验
4	1570.0	11500.0	经验不限	157983380976186	8000	无经验

图 5-7 初步表格数据

通过查询应聘岗位，发现应聘者中想要应聘数据分析师共有 10843 人

我们提取了‘平均工资、工作岗位要求经验’和‘最低薪资期望、工作经验’两者相合并形成一列新的列名命名为‘xdate、ydate’，为后续的岗位匹配度便于计算，将‘求职者 ID、招聘信息 ID、xdate、ydate’形成一个新的表格，为后续的文本训练做数据准备。部分结果展示如下图图 5-8 所示：

	求职者ID	招聘信息 ID	xdate	ydate
0	163561035761778	1590.0	12500.01年工作经验	4000无经验
1	163401909257279	1580.0	8000.01年工作经验	5000无经验
2	163294781611455	1580.0	16000.01年工作经验	70001年工作经验
3	163228658690333	1570.0	11500.0无经验	6000无经验
4	157983380976186	1570.0	11500.0无经验	8000无经验

图 5-8 准备数据

我们使用 jieba 工具对所得列 xdate、ydate 进行分词，使用自定义的词典，便于我们后续的模型训练，同时新建空列命名为 label，放置训练结果数值，对所得分词进行词频统计，数据展示表 1-3 词频统计：

词语	次数
经验	532
12500.01	40
工作	434
年	434
分析师	1284
不	922
无	98

8500.01	32
12500.03	28
20000.01	28
22500.03	24
17500.03	24

表 1-3 词频统计

在处理出使用的数据后，我们随机提取两个求职者信息做分词匹配测试，如使用 sent1 = "12500.01 年工作经验"、sent2 = "15000 无经验"作为测试用例，测试结果显示：两个字符串之间的匹配度为"0.8509632349014282
"，即该求职者所具有的技能经验和对岗位的要求工资等信息与该岗位需要具备的内容相关性高。

数据利用 collections 库中的 Counter，对分词的文本使用数值相替换，会更加有利于后续的满意度计算，通过对词频统计对各项词语进行数值转换，转换后的数据表，label 列为空，提前建立一个空列放置计算后的满意度数值，如下图所示图 5-10（工作 id 为招聘信息 ID，人才 ID 为求职者 ID）：

	工作id	xdate	人才id	ydate	label
0	1.585550e+18	[6, 3, 4, 2]	1635610357617786880	[1, 5, 2]	0
1	1.582579e+18	[25, 3, 4, 2]	1634019092572798976	[1, 5, 2]	0
2	1.578673e+18	[16, 3, 4, 2]	1632947816114552832	[1, 3, 4, 2]	0
3	1.574931e+18	[26, 5, 2]	1632286586903330816	[1, 5, 2]	0
4	1.569585e+18	[26, 5, 2]	1579833809761861632	[1, 5, 2]	0

图 5-10

使用的数值为文本分词后的词频统计对文字进行数值替换，对替换下来的数值进行模型训练，求职者满意度的主要的核心代码如下图所示图 6-7 部分核心代码：

```
while True:
    # 数据一轮循环(epoch)完成，打乱一次顺序
    if batch_count * batch_size + batch_size > data_size:
        batch_count = 0
        if shuffle:
            p = np.random.permutation(data_size)
            all_data = [d[p] for d in all_data]
    start = batch_count * batch_size
    end = start + batch_size
    batch_count += 1
    batch_data = [d[start: end] for d in all_data]
    batch_sent1, batch_sent2, batch_label = batch_data

    batch_sent1_pad = pad_sequences(batch_sent1, maxlen=max_len, padding='post')
    batch_sent2_pad = pad_sequences(batch_sent2, maxlen=max_len, padding='post')
```

图 6-7 核心代码

数据匹配计算满意度时，随机生成打乱的索引，然后再组织数据计算，与此循环，计算最终的结果，将具体得出的数据存储到数据表中，训练时采用了二八分原则，训练结果能达到百分之 80 以上，则使用改模型以及数据进行训练；为每位求职者提供求职者满意度非 0 的招聘信息，将结果进行降序排序存放保存在 result3-2 表中，部分结果如下图截图图 5-8 满意度训练结果：

	求职者ID	招聘信息 ID	xdate	ydate	label
6	163149102058582	1570.0	10000.01年工作经验	7000无经验	0.918020
12	163081854617295	1540.0	10000.01年工作经验	8000无经验	0.918020
15	161457598530964	1530.0	9000.01年工作经验	8000无经验	0.915210
8	163085245116488	1560.0	9000.01年工作经验	6000无经验	0.915210
9	163112663632799	1560.0	9000.01年工作经验	5000无经验	0.915210

图 5-8 满意度训练结果

6. 招聘求职双向推荐模型

6.1 岗位匹配度的数据处理

由第三问结果得知，企业发布的招聘岗位为 1574 条数据，人才网注册的用户为 19908 条数据，根据清洗与建模的数据训练，最终获取到的有效数据为企业发布岗位 1574 条，人才注册实名数据也是 1574 条，但在建模训练数据的过程中，发现对于 1574 个用户对应 1574 个岗位所生成的 2,477,476 条，接近两百五十万条数据在该计算机的运行中不堪重负，即使在后端开设了 16 个进程进行分流和较

少内存的运行，也很难准确跑完所有数据，所以从第三题的建模分析开始，就选取了前 700 个注册有效用户进行数据训练，即将近一百万条数据进行训练和分析，第四题也是延续第三题做的分析，因此也是前 700 个有效用户的数据，作为我们双向推荐模型的训练对象。

提取原始表中的 result1-1.csv 中的工作岗位招收人数，与 result3-1.csv 表格数据相合并，将工作岗位招收人数整合到数据表中，获取需要分析的数据集合，形成一个表格数据，为后续的招聘流程提供相关数据。

根据要求对匹配度进行非‘0’，即‘1’的转化筛选，需要对所有的匹配值求平均值，匹配值高于或等于平均匹配度设置为 1，低于平均匹配度设置为 0。

首先查看匹配度的平均值为'0.4262311109612694'，之后查看每一个匹配度值均为'float'类型，即数值型，且平均值也是数值型数据，在做 for 遍历不需要转型可以直接运算，转换完查看得知满意度中 0 值有“499504”个数据，满意度中 1 值有“444896”个数据，下图代码为匹配度转化筛选的代码，图 6-1。

```
pp_tf = []
a=0
b=0
for i in pipei['label']:
    if i < pp1:
        pp_tf.append(0)
        a += 1
    else:
        pp_tf.append(1)
        b += 1
print('满意度中0值为',a)
print('满意度中1值为',b)
```

```
满意度中0值为 499504
满意度中1值为 444896
```

图 6-1

运算后得出的列表并入原数据中，得知总共有"1"值"4999504"条数据，"0"值"444896"条数据。

下图为岗位匹配度的数据集合展示：

	工作id	人才id	工作岗位招收人数	label	label_tf
0	1.482200e+18	7.539530e+18	2	0.867432	1
1	1.482200e+18	7.539480e+18	2	0.867432	1
2	1.482200e+18	7.539530e+18	2	0.862260	1
3	1.482200e+18	7.539480e+18	2	0.862260	1
4	1.482200e+18	7.539480e+18	2	0.862260	1
...	...	...	...	...	...
944395	1.494560e+18	7.539740e+18	2	0.005637	0
944396	1.494560e+18	7.539740e+18	2	0.005008	0
944397	1.494560e+18	7.539740e+18	2	0.005008	0

图 6-2 岗位匹配度数据集合

6.2 求职者满意度的数据处理

根据要求对满意度进行非‘0’，即‘1’的转化筛选，需要对所有的满意值求平均值，满意值高于或等于平均匹配度设置为 1，低于平均满意度设置为 0，查看满意度的平均值为’0.8494644821586466’，转换完查看得知满意度中 0 值有“121”个数据，满意度中 1 值有“145”个数据，具体代码如下图所示图 6-3 求职者满意度的数据集合处理：

```
my_tf = []
c=0
d=0
for i in myl:
    if i < my_agvx:
        my_tf.append(0)
        c += 1
    else:
        my_tf.append(1)
        d += 1
print('满意度中0值为',c)
print('满意度中1值为',d)
```

```
满意度中0值为 121
满意度中1值为 145
```

图 6-3 求职者满意度的数据集合处理

下图为求职者满意度的数据集合展示：

工作id	xdate	人才id	ydate	label	my_tf
1.482196e+18	20000.05年工作经验	1507237125567938560	60001年工作经验	0.871785	1
1.482196e+18	12500.03年工作经验	1470311341238648832	90001年工作经验	0.868453	1
1.482192e+18	12500.03年工作经验	1462685278807392256	110001年工作经验	0.868453	1
1.482195e+18	20000.03年工作经验	1484369311995920384	80004年工作经验	0.866984	1
1.482196e+18	40000.05年工作经验	1461666278635864064	150001年工作经验	0.866243	1
...	...	...	...	...	...
1.482185e+18	12500.0无经验	7539868482224713236	6000无经验	0.834278	0
1.482195e+18	6000.0无经验	1470313042016337920	8000无经验	0.831967	0
1.482195e+18	6000.0无经验	1472772112841310208	9000无经验	0.831967	0

图 6-4 求职者满意度数据集合

图片中的 my_tf 例放置了求职者满意度。

6.3 offer 的发行以及履约率

本题赛题描述的招聘流程如下：设某岗位拟聘 n 人，泰迪内推平台向企业推荐岗位匹配度非 0 的 n 位求职者发出第一轮 offer，求职者如果收到多于 1 个岗位的 offer，则求职者选取满意度最高的岗位签约，每个求职者只允许选择 1 个岗位签约。第一轮结束后，平台根据当前各招聘信息的剩余岗位数，向后续被推荐求职者发出第二轮 offer，如此继续，直到招聘人数已满或者向所有拟推荐求职者均已发出 offer 为止。履约率定义为：履约率=所有岗位的签约人数之和/所有拟聘岗位人数之和。

第一轮的 offer 发送，企业端：企业岗位招收人数非 0 的岗位向匹配度值为“1”的人才 id 发 offer，对应满意度也为“1”的求职者设置 offer 一栏的数据为 true，同时企业工作岗位招聘人数减 1，求职者端：由于热门的企业岗位大家都会去投递，而优秀的求职者也有很多企业都发了 offer，但在此途中求职者只能选择最多其中一份岗位去入职，即接受其中一份岗位，且前一位求职者接受的岗位，如果该岗位招收人数没有空缺，那么下一位求职者不能就职这个岗位，只能选择下一个别的岗位，因此，在对第一次企业发了 offer 之后，部分 offer 被设置为 true，但仅仅代表这个岗位对符合条件的求职者发了 offer，并不是求职者也确定了 offer，接下来还要去做第二次的求职者的反向选择。

对企业招聘岗位中的招聘人数进行空值处理，查询得出共有 64 个岗位并没有开始招聘，将工作招收人数为 0 的数据设为空值，空值代表着岗位并不需要招聘人才，所有综合考虑我们对空值的相关岗位进行一次性删除空值，得到剩下的 843664 条数据，对数据重新设置索引，遍历查询工作表中匹配值为“1”的数据，判断如果求职者表中的满意也为 1，即设置 offer 一栏的数据为“true”。具体结果如下图所示图 6-5 重置后结果：

	工作id	人才id	工作岗位招收人数	label	label_tf	offer
0	1522090000000000000	7539910000000000000	3	0.732133	1	True
1	1522090000000000000	7539910000000000000	3	0.732133	1	True
2	1522090000000000000	7539910000000000000	3	0.732133	1	True
3	1522090000000000000	7539910000000000000	3	0.732133	1	True
4	1522090000000000000	7539910000000000000	3	0.732133	1	True
...	...	...	...	...	...	...
843659	1494560000000000000	7539740000000000000	2	0.005637	0	False
843660	1494560000000000000	7539740000000000000	2	0.005008	0	False

图 6-5 重置后结果

当求职者查看收到 offer 的求职者总共有 424980 条数据，即为 270 名求职者收到了第一轮的 offer，第一轮 offer 结果如下图图 6-6 第一轮 offer 结果显示：

	工作id	人才id	工作岗位招收人数	label	label_tf	offer	offer2
0	1522090000000000000	7539910000000000000	3	0.732133	1	True	False
1	1522090000000000000	7539910000000000000	3	0.732133	1	True	False
2	1522090000000000000	7539910000000000000	3	0.732133	1	True	False
3	1522090000000000000	7539910000000000000	3	0.732133	1	True	False
4	1522090000000000000	7539910000000000000	3	0.732133	1	True	False
...	...	...	...	...	...	...	...
843659	1494560000000000000	7539740000000000000	2	0.005637	0	False	False
843660	1494560000000000000	7539740000000000000	2	0.005008	0	False	False
843661	1494560000000000000	7539740000000000000	2	0.005008	0	False	False

图 6-6 第一轮 offer 结果

履约率定义为：履约率=所有岗位的签约人数之和/所有拟聘岗位人数之和，计算在岗签约人数、拟聘岗位人数以及履约率，可见第一轮的 offer 签发中履约率只有“0.00012706480304955527”，计算代码如下图 6-7 履约率代码：

```
len(ab[ab['offer2'] == True])
len(ab[ab['offer'] == True])
lvy1 = len(ab[ab['offer2'] == True])/len(ab[ab['offer'] == True])
lvy1
```

图 6-7 履约率代码

由于第一轮的履约率只有 0.00012706480304955527 极低，为了提高企业岗位发放 offer 的履约率，进行训练代码的调优，要使得履约率达到最高，但是招

聘岗位数量是固定的，只能尽可能减少收到 offer 的人数，即通过准确的发出 offer，把 offer 发给意愿更高的求职者。

在发放 offer 的同时先了解求职者对于这个岗位的满意程度，且这个求职者有没有对其他岗位的满意程度也达到了 1，按照对比择优原则，选择“满意程度为 1”的数量尽量少的求职者发放 offer，而不是对所有满足条件的求职者都发放 offer，则是在设置 offer 为 True 之前添加筛选条件，达到双向奔赴最大可能性的训练调优。具体代码如下图图 6-8 训练代码所示：

```
t_offer = []
ab = []
label_id = []
for q in manyi:
    if q['label'] = 1:
        label_id.append(q['人才id'].index.to_list())
        for i,f in zip(pipei['人才id'],range(0,len(pipei['人才id']))):
            if i == q['人才id'][f]
                pipei_tf = pipei[pipei['人才id'] == i]
                for j in range(0,len(pipei_tf)):
                    a.append(i)
                    if pipei_tf['offer'][0] == True: # 判断该id中第一个值为True
                        t_offer_j = []
                        t_offer_j.append(True)
                    else t_offer_j.append(False)
                tt_offer = {
                    a[j]:t_offer_j
                } # 做列表合并
pipei['t_offer'] = tt_offer
```

图 6-8 训练代码

调优后的履约率提高了 2.2 倍由于第二轮、第三轮的 offer 发放与第一轮大同小异，数据源本体也没有发生变化，数据的训练结果几近相同

参考文献

[1]李玉志[1]，刘晓亮[1]，邢方方[1]，温国强[1]，卢楠滟[2]，何慧[2]，焦润海[2] 基于 Bi-LSTM 和特征关联性分析的日尖峰负荷预测 10.13335/j.1000-3673.pst.2020.1390

[2]吕乐宾 刘群 彭露 邓维斌 王崇宇 结合多粒度信息的文本匹配融合模型 DOI：10.11896/jsjkx.200700100

[3]郑思雨. 网络招聘信息的数据挖掘研究[D].杭州电子科技大学,2020.DOI:10.27075/d.cnki.ghzdc.2020.000039.