# 金融产品指导师——智能小投

组长：邱俊源

组员：张凡、曾旭祥、杨继超、邵典、范一迪

## 一、背景研究

2018年4月《关于规范金融机构资产管理业务的指导意见》（以下简称“资管新规”）发布以来，为各大资管机构（主要代指银行）培育正规理财产品提供了宽松的政策条件。条款发布后的次年已然成为我国银行系理财子公司的设立元年。据统计，目前 6 家国有大行理财子公司已全部开业，12 家全国性股份制银行中已有 10 家完成理财子公司的获批筹建，从**政策和市场供应者角度**推进了各类理财产品的蓬勃发展；而从**市场需求角度**来看，互联网金融理财产品也对经济、金融具有较大影响，随着互联网金融理财产品在大众中被认可度的提升，为迎合市场需求，越来越多的金融理财产品进入了大众视野。

金融理财产品种类各异，数量众多。如各类互联网平台，包括（新浪微财富、网易现金宝等）、电商（阿里巴巴余额宝、腾讯理财宝、京东小金库等）、搜索网站（百度百赚、百发等）发布的互联网理财产品；基于各类货币基金产品发布的理财产品，例如华夏现金增利货币基金推行的活期通、汇添富基金全额宝等；各大银行发布的各银行自身的系列理财产品（如工商银行薪金宝、平安银行平安盈、中国银行活期宝等）；电信运营商主推出的财务余额自动理财类服务，如中国电信翼支付联合民生银行推出的“添益宝”等。

但与此同时，伴随着金融理财产品种类与数量的日益增长随，消费者对理财产品的认知度并未提高，尤其对银行理财的一些专业术语一知半解，对于一些理财新手来说，更是难上加难。繁杂的理财产品种类与金融产品大量且晦涩的专有名词是大多数普通消费者面临的一项**信息壁垒**。如何在繁复的资源中获取各类理财产品的精准信息，并准确理解各类理财产品晦涩的专有名词，获取各种理财产品的性质和属性，已经成为大众面临的主要问题之一。

## 二、数据来源和获取

### 2.1 数据来源

说明书和风险揭示书的PDF均来自于招商银行相关网站[[1]](#footnote-1)。一共爬取了632份理财产品《说明书》和623[[2]](#footnote-2)份理财产品《风险揭示书》。由于时间和项目进度的关系，同时也因为部分说明书排版和文字上的差异过大，**为了达到项目“最小可用”的状态，我们共同商议后决定先选取《说明书》排版格式类似的44份理财产品作为“第一代”产品上线**。考虑商用后，可以在协调产品提供方提供较为标准化的产品《说明书》和《风险揭示书》的基础上扩展分析的PDF的模板，以增加数据的多样性，更好满足客户的需求。

### 2.2 数据获取（邱俊源）

在Jupyter Notebook上使用python爬虫来获取理财产品《说明书》和《风险揭示书》。以下将详细阐述具体的重要的步骤和技术。

1、考虑招商银行网站属于“下滑后再加载”的网站，于是我们使用selenium模拟浏览器的滑动行为，待页面的完全加载后再进行相关的爬虫以保证信息的全面性。

2、为了防止触发网站的反扒机制，我们使用了不同的headers进行信息的获取，并配合短暂的停顿，使我们的爬虫程序能够进行完整的爬取。

相关函数说明如下表所示。

|  |  |
| --- | --- |
| 函数名称 | 作用 |
| get\_user\_agent\_pc get\_user\_agent\_phone | 分别模拟PC端和手机端对网站进行访问 |
| point\_to\_page | 直接跳转到相应的页面 |
| get\_file | 根据URL下载对应的PDF |
| get\_pdf | 爬虫程序主体 |

## 三、信息提取

### 3.1 文档表格提取（邱俊源）

通过观察我们发现，产品《说明书》文档中均存在多个表格，并且表格内的信息也大部分涵盖了该理财产品说明书的主要内容。于是，我们首先尝试提取理财产品《说明书》中的所有表格的信息。以下将详细阐述具体的重要的步骤和技术。

1、首先，我们使用python调用电脑的Adobe Acrobat软件将所有的《说明书》PDF转换为XML格式文件，以方便表格的识别和内容的提取。相关函数说明如下表所示。

|  |  |
| --- | --- |
| 函数名称 | 作用 |
| get\_list\_files | 获取目录下的所有文件名 |
| pdf2xml | 将一个PDF转换为XML |
| change\_all\_files\_to\_xml | 批量转换PDF为XML |

2、之后，我们根据XML中表格的标签（<table>）定位表格并读取表格的内容写入DataFrame中。根据两个表格标签的临近程度来判断是否是因为分页而导致的表格拆分。如果是，则对这些表格进行合并处理。相关函数如下表所示。

|  |  |
| --- | --- |
| 函数名称 | 作用 |
| transfer\_xml\_table\_to\_pandas | 将一个<table>标签写入DataFrame中 |
| get\_all\_tables | 将所有PDF中的所有表格进行转换为PDF并依据标签位置进行连接 |

3、根据表格合并结果对表格的内容进行合并，完全解决表格分页的问题。相关函数如下表所示。

|  |  |
| --- | --- |
| 函数名称 | 作用 |
| revise\_break\_off\_table | 将合并后表格的内容进行修正 |

**鉴于仍旧存在部分表格缺失、不一致的情况，我们选取前2个表格[[3]](#footnote-3)作为理财产品的信息输入。**

### 3.2 表格信息提取（邱俊源）

从PDF中提取完两个表格的信息后，我们对表格中的数据进行了相应的处理，以便更好地进行后续的存储和查询操作。以下将详细阐述具体的重要的步骤和技术。

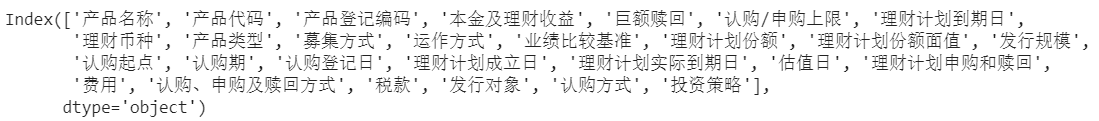
1、首先是对理财计划要素表进行了处理，主要处理操作有：去除空格、分离聚合的信息、修改不一致的说明等。相关函数说明如下表所示。

|  |  |
| --- | --- |
| 函数名称 | 作用 |
| clear\_blank | 去除不必要的空格 |
| extract\_info\_from\_table\_1 | 对理财产品要素表的信息进行提取 |

2、之后是对投资比例表进行了处理。因为该表涉及一个投资品种之间包含和并列的层级关系，目前我们还没找到比较好的能够精准、清晰地提取出这种内在关系的算法。所以出于对信息提供的准确性和可靠性的考虑，我们选择直接将表格内容呈现给顾客。相关函数说明如下表所示。

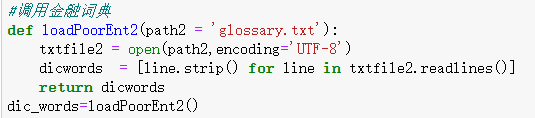
|  |  |
| --- | --- |
| 函数名称 | 作用 |
| extract\_info\_fro\_table\_2 | 对投资比例表的信息进行简单的清理并转化为HTML文本以便后续展示 |

最终，我们从这44份理财产品《说明书》的表格中提取出了如下28个字段，其中27个字段来自于理财产品要素表，1个字段来自于投资比例表。

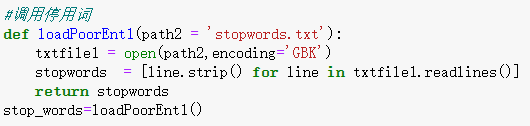


### 3.3 文档金融专有词汇提取（邵典、范一迪）

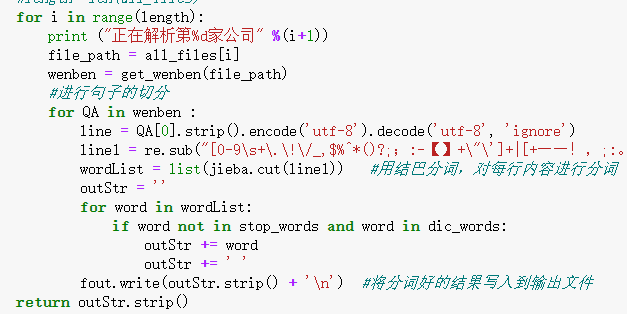
考虑金融领域词汇的特殊性，使用金融领域词典作为分词基础：



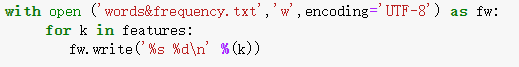
为了全面、精确解释金融理财产品中出现的各类专有名词，加入停用词去除一些日常词汇：



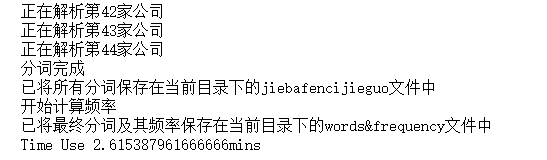
通过使用正则表达式将数据进行分词：



将分词结果以.txt文件格式输出到目录下：



输出结果如下：



### 3.4 金融专有词汇说明提取（邵典、范一迪）

MBA智库3是一个专注于经济管理领域的知识平台，能够在保证专业性的同时提供简单易懂的金融词汇解释，因此选择MBA智库百科作为词汇解释的爬取对象。

首先明确爬虫的目标，是希望能够将说明书和风险揭示书中提取的专有名词词条在MBA智库百科进行针对性检索并将获取到的解释存储到本地，获取到的解释数据的宗旨是取精去糟，通俗精练，不要拿到无关信息（如广告与推荐各种无关信息）。具体操作步骤如下：

首先导入fake\_useragent库，fake\_useragent库是专门构造伪装请求头的库用来构造随机请求头，相对于使用固定头更加保险。然后定义请求头，在后续访问网址的时候，需要在url后面加入请求头信息。

读取说明书和风险揭示书nlp自然语言处理后提取出的金融词典（words&frequency.txt）4，金融词典以txt格式存储，格式如图所示

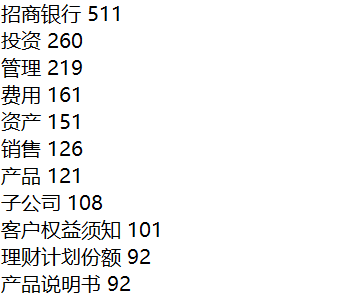


图1.2 金融词典（words&frequency.txt）存储格式

图中显示信息部分由两部分组成，分别为自然语言处理得到的金融词汇和对应的词频，后续的查询将以词频大小为顺序，对词典中读取得到的词汇解释进行爬取与存储。

获取MBA智库百科词条查询的url。（提示由于url均为固定url前缀+查询词，因此采用直接生成url的方式）

使用Beautiful Soup库，进行网页抓取数据。

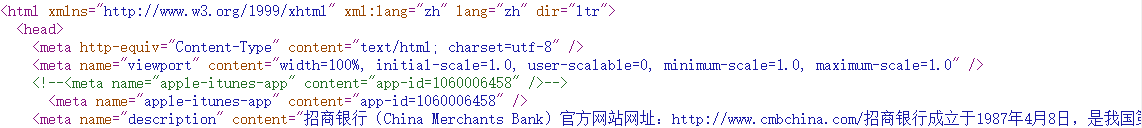
以商业银行为例（url=https://wiki.mbalib.com/wiki/商业银行）

审视页面



可以明确我们的爬取目标为招商银行简介对应部分

右键查看看页面源码



由页面源码可知，通过name定位至'description'位置，获取content即为该词条解释。

设定时间间断5s以解决爬虫被封的问题，若不设定间断，ip将在连续100个词条左右后被封。

## 四、信息录入数据库

### 4.1 将产品属性写入Neo4j数据库中（邱俊源）

***为什么要使用Neo4j数据库进行存储而不是传统的关系型数据库？***

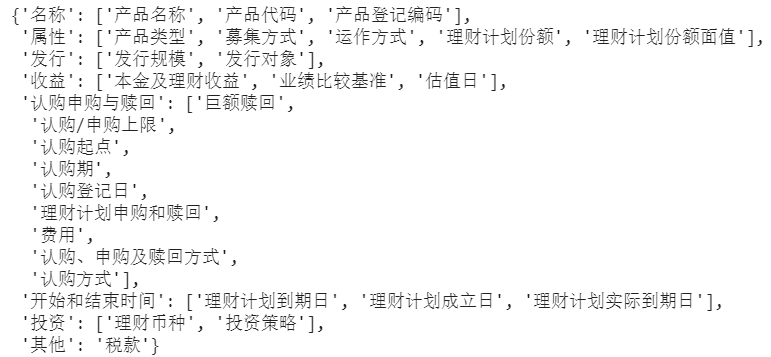
1、首先，使用图的方式来表达现实世界的关系**很直接、自然，易于建模**。在我们的产品中，存在着两种类型的节点，一种是理财产品实体节点（名称、编号），另一种是理财产品属性节点（认购起点、理财币种等）。一个理财产品实体可以和多个不同的属性节点相连，一个属性节点也可以有多个相连的理财产品实体，这就构成了理财产品网络。

2、其次，图数据库可以很高效的插入大量数据。考虑到后续商用，**产品的可扩展性也是十分重要的**。传统的关系型数据库，例如MySQL，在不做分表分库的情况下插入百万数据基本的插入和查询就已经很慢了。而图数据库基本能胜任亿级以上的数据，持续插入十亿级的数据基本还能保持一个较高的插入和查询速度。

3、最后，**图数据库可以很高效的查询关联数据**。比如，一位用户想查询如下内容：请告诉我在青葵系列理财产品中投资币种为美元的那些产品各自的投资策略是什么。这就涉及到了一个多层关联和筛选的过程。传统的关系型数据库一般来说都需要做表连接，而表连接是一个很昂贵的操作，涉及到大量的IO操作及内存消耗。图数据库对关联查询一般都进行针对性的优化，比如存储模型上、数据结构、查询算法等，防止局部数据的查询引发全部数据的读取。

综上所述，我们决定采用Neo4j数据库对理财产品属性进行存储和查询。以下将详细阐述具体的重要的步骤和技术。

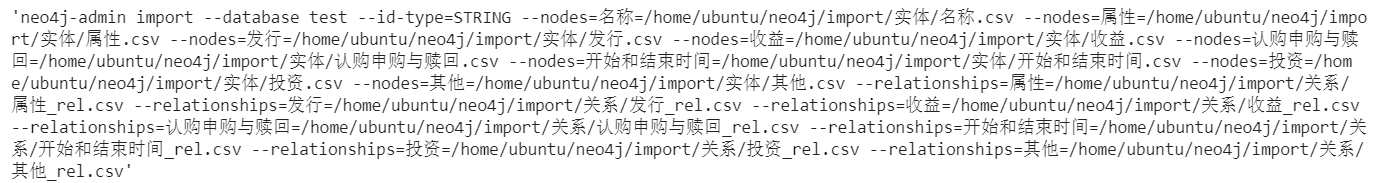
1、将理财产品的属性进行归类，分别存入不同类型的节点中，并以name和value两个属性存储以方便后续的查询。并且还提取除了日期、收益率等可以比较的理财产品属性，将其存储在新的“up”和“down”等节点属性中，具体的分类情况如下图所示。



2、将理财产品属性加入图数据库中。参考这篇博文[[4]](#footnote-4)，我们使用千万级别节点的导入方式neo4j-import方式批量导入的方式将数据导入图数据库。这样做的好处是可以快速、成规模地导入数据，而不是运行cypher语言进行插入，可以更好地为之后的商用环境做准备。首先，我们得先为每一个节点建立一个独一无二的ID，之后根据这个ID分别构建每一个实体类的实体表和实体之间的连接表。最后通过命令行的方式进行快速导入。相关函数说明如下表所示。

|  |  |
| --- | --- |
| 函数名称 | 作用 |
| create\_search\_df | 为每个节点建立独一无二索引ID |
| create\_node\_csv | 建立实体表 |
| create\_relationship | 建立关系表 |

使用python自动生成命令行语句并复制到命令行运行。



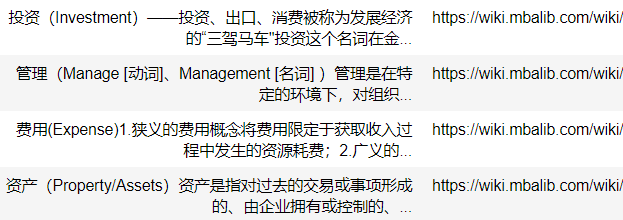
最终生成了324个节点和968条关系，如下图所示。

### 4.2 将金融专有名词词典写入MySQL数据库（范一迪）

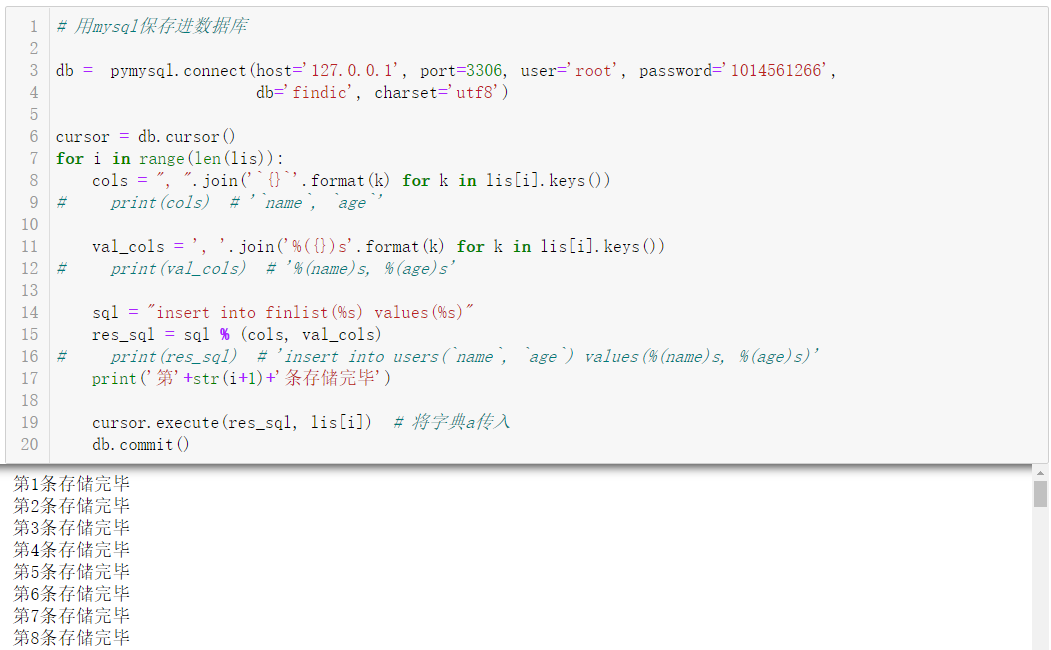
通过python进行的爬虫与分析清洗操作，我们已经实现了对金融名称数据的获取与预处理。

接下来我们要通过mysql和python的交互，将数据写入MySQL数据库中：

首先看一下数据的df：



现在要将这个df写入mysql，也就是把爬到的数据或者清洗完的数据写入mysql，具体关键代码如下：



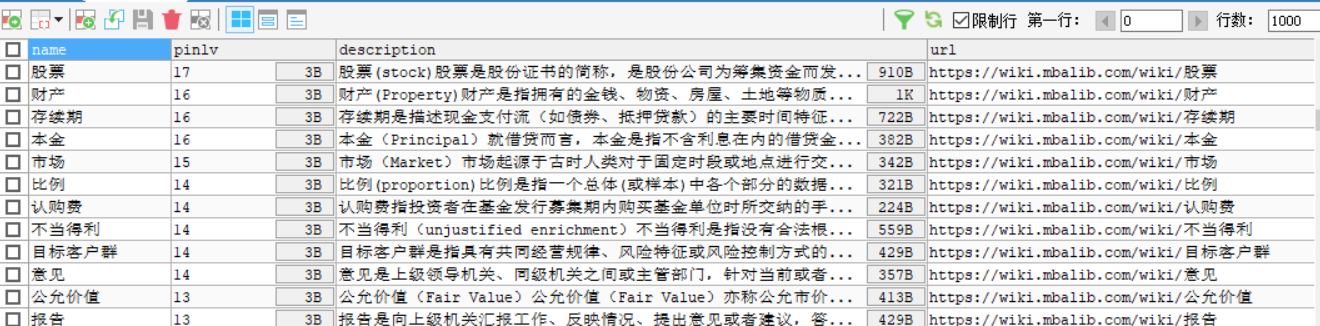
......全部数据存储完毕

初次运行过程中捕获到了异常：



这是由于数据中的一部分拉丁文无法编码，设置一下编码方式charset=utf8即可解决

存储完毕后我们打开本地的数据库，右键刷新即可看到：



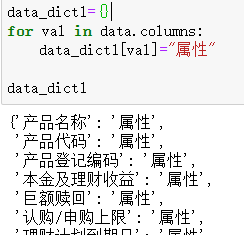
数据已经成功写入mysql了

## 五、客户提问NLP分析

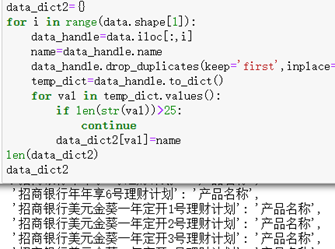
### 5.1 基于正则表达式的NLP（邵典、曾旭祥）

### 5.1.1 用户输入分词

读入已经经过处理的excel文件并将其转化为字典类型，分别有{属性名：'属性'}和{元素：属性名}字典：

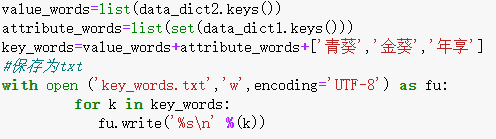


图：{属性名：'属性'}字典



图：{元素：属性名}字典

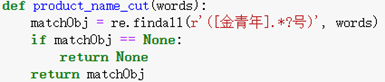
考虑到pdf中出现的特有名词不会被过滤掉，加入考虑两个字典键值内容的分词文件并保存为key\_words.txt：



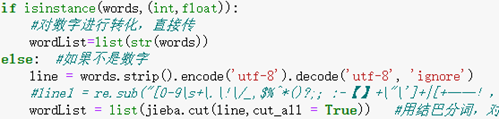
为了更好的区分不同的名词，分词函数考虑“的”，“是”这样的助词：



并且将此系列名词[金青年].\*?号作为名词考虑到分词中：

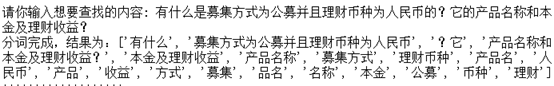


之后采用和pdf提取类似的分词方式对用户输入语句进行分词，不同于pdf分词，这里参数直接传入字符串，并且考虑用户输入数字的可能性，在用分词词典和专有词典对字符串进行切分分词：



如下为输入后输出示例：





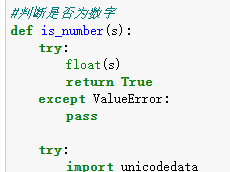
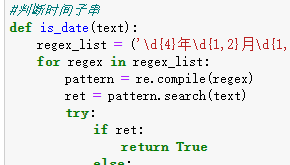
### 5.1.2 NLP主体、属性值识别及相关处理

search\_detail(user\_input)是nlp语义识别的主函数，通过调用分词、模糊匹配、多主体识别、多属性识别等函数，获得最终需要查询的结果，部分截图如下。



其中输入进函数中分词后的词组需要被判断是否属于数字或日期，如果是则进行特殊处理：

判断分词词组中某个索引的值，使用is\_number(s)方法和is\_date(text)进行判断，以下为部分截图。

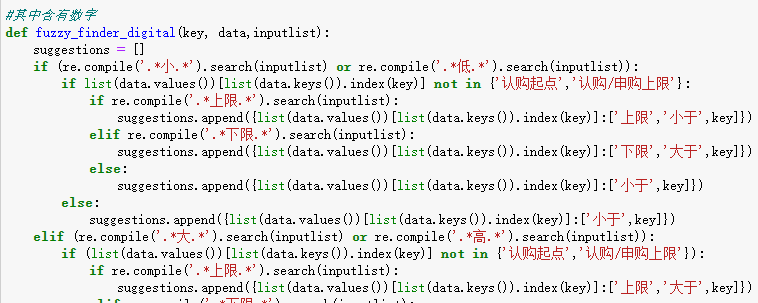
 

判断分词词组中索引值属于哪一类后，其中用三个方法来进行分词结果的主体识别。分别处理纯文本索引值，日期类型的索引值和数据类型的索引值。

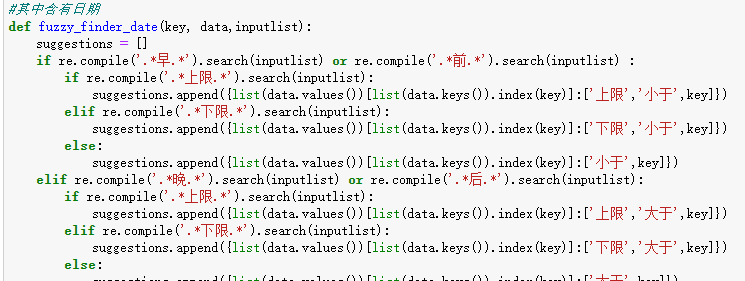
fuzzy\_finder\_subject(key,data,inputlist)方法是用来处理纯文本的方法，主要进行分词结果的主体识别，采用模糊匹配的方式，对关键词key进行匹配，添加判断，排除无关的结果，最终得到主体的输出。



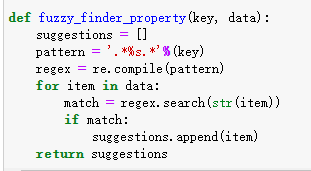
fuzzy\_finder\_digital(key,data,inputlist) 方法是用来处理含有数字的语句的方法。主要通过分析语句中含有的特定词和限制来对数字部分添加相关条件，特定词有“小”，“低”，“大”，“高”等。,若语句中含有以上词汇，则会输出特定格式的主体内容。以下是部分截图：



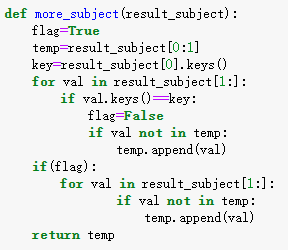
fuzzy\_finder\_date(key,data,inputlist) 方法是用来处理含有日期的语句的方法。主要通过分析语句中含有的特定词和限制来对日期部分添加相关条件，特定词有“早”，“晚”，“前”，“后”等。,若语句中含有以上词汇，则会输出特定格式的主体内容。以下是部分截图：



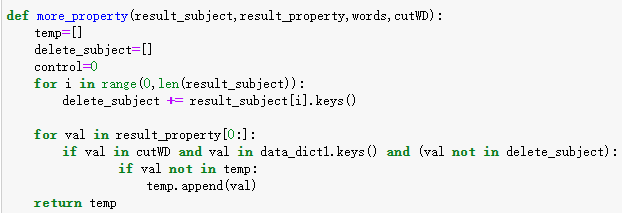
fuzzy\_finder\_property(key,data)方法主要进行分词结果的属性值识别，利用正则表达式进行模糊匹配，得到属性值的输出。



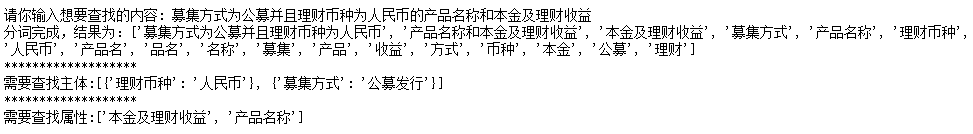
more\_subject(result\_subject)方法剔除主体中的重复结果，以及识别出多个需要查询的主体。

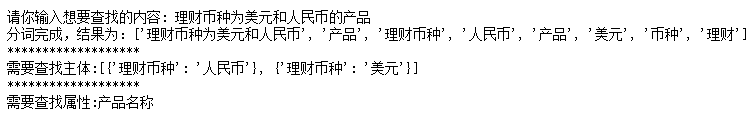


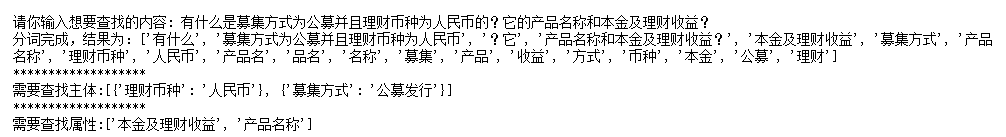
more\_property(result\_subject,result\_property,words,cutWD)方法识别出多个需要查询的属性值。



测试语句的输出如下：







为了匹配后续查找，对输出的格式进行更改，首先我们要规范化输出的格式如下：



按照这个格式，我们把'产品名称', '产品代码', '产品登记编码'这三个从输出的查找主体中分离出来作为‘Object’里的数据，其余的留在‘Attribute’中。‘Return’中的值即为需要查找的属性值。

将未规范前的语句传入clea\_re\_result(result\_1,result\_2)方法，此方法为清洗核心方法，传出值即为最终清洗后的值。方法内首先对输出主体进行拆分，得到‘Object’，部分截图如下。



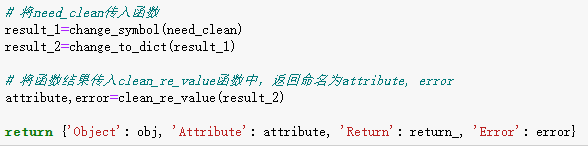
再通过change\_symbol(need\_clean)方法和change\_to\_dict(result\_1)方法将含有数字和日期的词规范化，将多个相同属性下的值放到一起并添加“或”字段，之后将输入整体从列表转化为字典形式，部分截图如下。



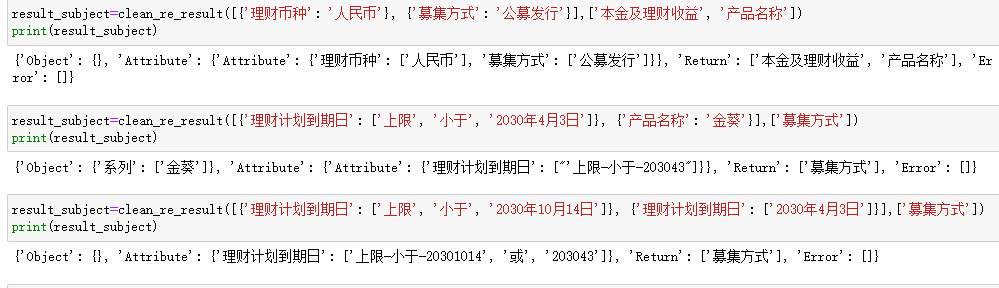
最后进行语义分析，分析语句中日期格式，数字格式和逻辑关系，部分截图如下。



三个方法在clean\_re\_result(result\_1,result\_2)方法中被调用，并输出结果：



测试输出如下：

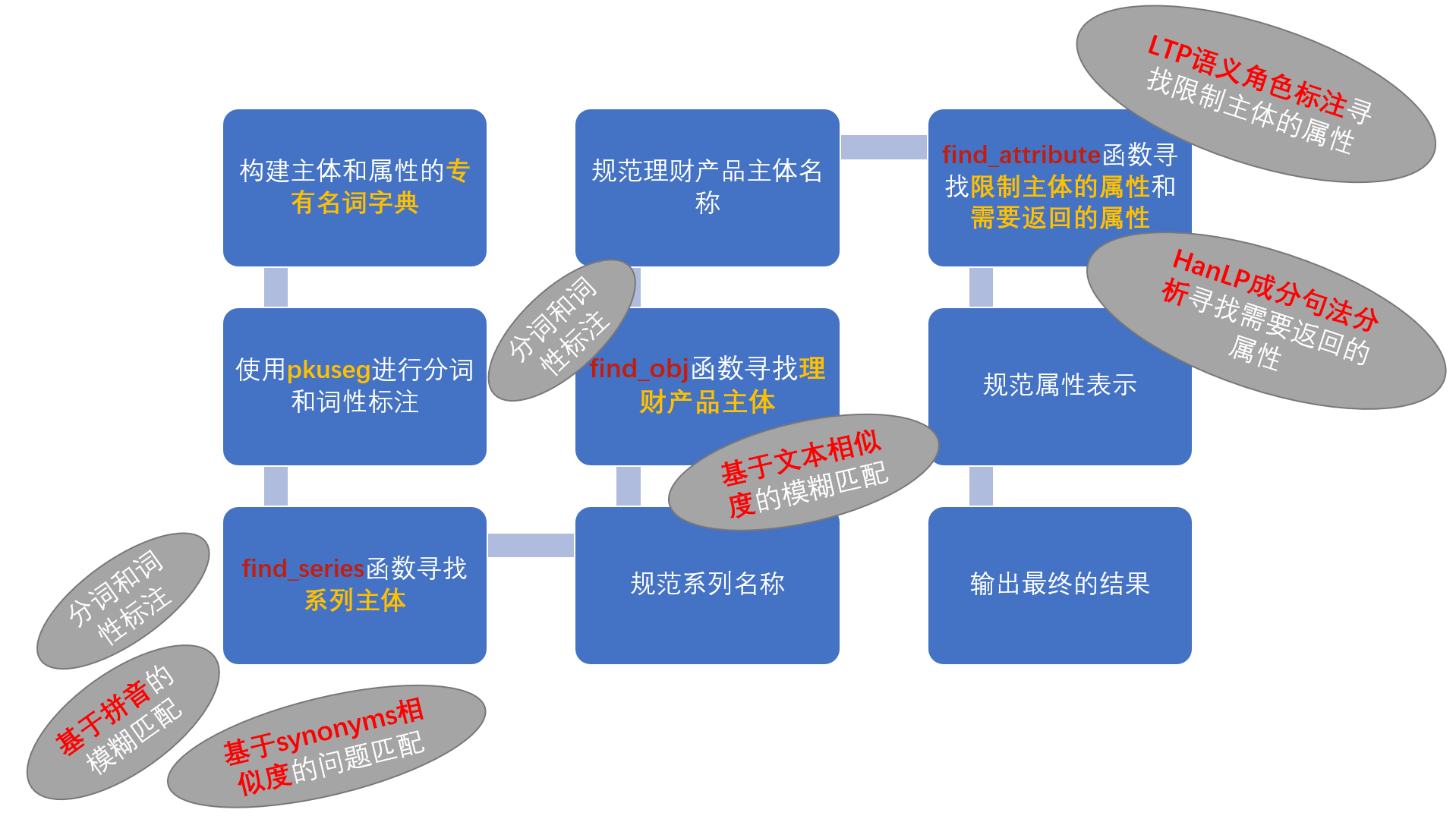


### 5.2 基于语义分析的NLP（邱俊源）

**语义分析**在本文中指的是不完全依赖正则表达式的强制匹配模式寻找主体和属性，而是通过使用专业的处理自然语言的python第三方库对用户提问句子的语法结构进行分析，从而提取出用户想要寻找的产品及属性。使用到的第三方库及简单说明如下表所示。

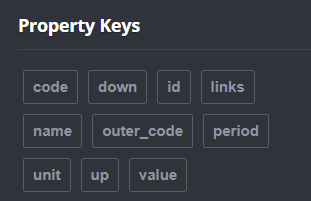
|  |  |
| --- | --- |
| 第三方库名称 | 简单说明 |
| LTP[[5]](#footnote-5) | 哈工大语言技术平台 |
| pkuseg[[6]](#footnote-6) | 北京大学开源分词工具 |
| HanLP[[7]](#footnote-7) | 前沿的开源自然语言处理包 |
| synonyms | 中文近义词工具包 |
| pypinyin | 汉字拼音转换工具包 |
| difflib | 文本对比工具包 |

下图为语义分析NLP的简要步骤说明



相关函数说明如下表所示。

|  |  |
| --- | --- |
| 函数名称 | 作用 |
| semantic | 用户提问解析主函数 |
| nlp\_return | 返回语义分析NLP结果 |
| find\_series | 查找系列实体 |
| find\_obj | 查找名称实体 |
| find\_attribute | 属性识别和归属 |
| similarity | 衡量两个字符串之间的相似性 |
| clean\_result | 对语义分析NLP结果进行规范 |

**1. 语义角色标注识别限制类属性**

语义角色标注是一种浅层语义分析技术，以句子为单位，分析句子的**谓词-论元**结构，其理论基础来源于Fillmore(1968)年提出的格语法，不对句子所包含的语义信息进行深入分析。具体来说，语义角色标注的任务就是以句子的谓词为中心，研究句子中各成分与谓词之间的关系，并且用语义角色来描述他们之间的关系。下面将展示一个简单的例子：



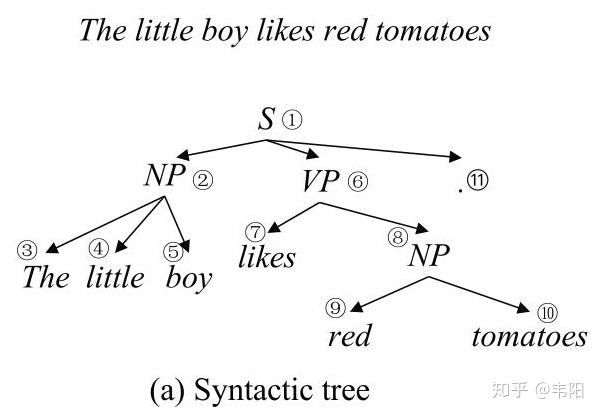
谓词 ：“发表”，施事：“奥巴马”，受事：“演说”，时间：“昨晚”，地点：“在白宫”

可以看到，语义角色标注就是要针对句子中的（核心）谓词来确定其他论元的角色。

在用户提问中，可能存在限制类属性，例如：收益率下限大于5%，到期日早于2020年，理财产品投资币种为美元或人民币等限制。我们发现，通过以谓词为中心的语义角色标注识，可以很好地识别出限制类属性的“动词”（之后转换为数据库查询中的符号——例如，早：小于，为、是：等于…）。因为客户提问的语句句法差异的问题，导致语义角色标注出的施事者（通常认为是需要加上限制的属性）和受事者（通常认为是该限制的属性的取值）与提问者的真实意图产生偏差。因此，我们在find\_attribute函数中对其进行了一些处理，针对目前发现的偏差类型进行了处理。

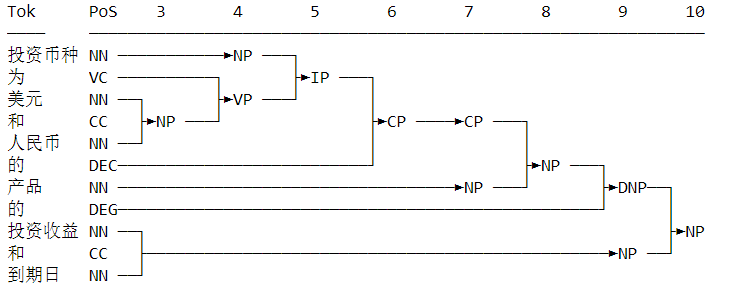
2. **成分句法分析识别需要返回的属性**

成分句法分析（constituent parsing）是自然语言处理中的一个基础任务，它的任务是给定一个句子，分析出句子的**短语结构句法树**。下面将展示一个简单的例子。

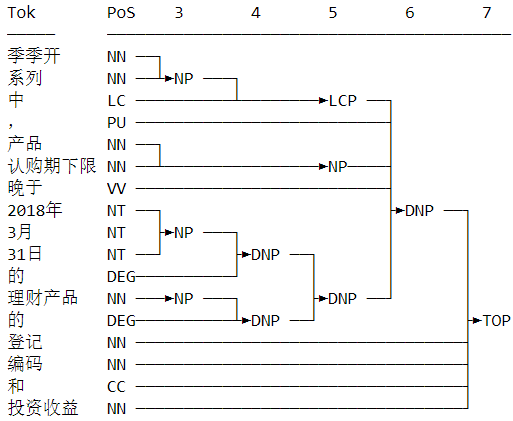


目前考虑了两种句法分析格式来识别需要返回的属性：

1、整个句子中的限制类属性可以被识别为DNP（修饰名词短语的结构）格式，需要返回的属性被识别为NP（名词短语）格式，如下图所示。此时，NP即为需要返回的名词。



2、整个句子的限制性属性部分可以被识别为DNP，但需要返回的属性由于句子结构和分词的问题不能被识别为NP。这时候我们取最后出现在层次结构中的DNP 结构作为分水岭，该DNP结构



## 六、数据库查询

### 6.1 Neo4j数据库查询（张凡、曾旭祥、杨继超）

### Object部分

### Attribute部分

这部分的输入是一个包含属性名称和属性值的键值对的字典，类似于，{“认购期”：[“上限-小于-20250101”，“并且”，“下限-大于-20180101”，“或”，“上限-小于-20210101”，“并且”，“下限-大于-20150101”]}，我们需要使用下面的process和greater\_or\_smaller函数来生成判断条件，然后传给query\_by\_attribute执行neo4j查询，返回实体的名称。

|  |  |
| --- | --- |
| 函数名称 | 作用 |
| query\_by\_attribute | 根据属性来查找实体名称。 |
| process | 将属性值转换为相应的查询判断条件 |
| greater\_or\_smaller | 比较符号的文字转符号 |

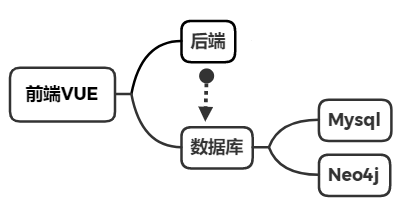
### Return部分

这部分的输入有两个，一个是ls\_names，代表实体名称的列表，即前两部分Object和Attribute部分的返回值的交集；另一个是ls\_attrs，代表用户想要查询的属性值的属性名称列表。遍历ls\_attrs，查询ls\_names中每个实体的对应属性值，这里面需要使用很多判断来改变查询的where条件。

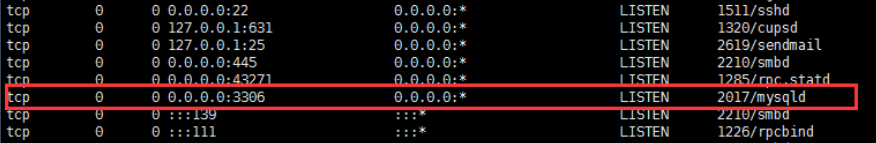
|  |  |
| --- | --- |
| 函数名称 | 作用 |
| query\_by\_name | 根据实体名称来查询用户想要查询的属性名的属性值 |

### 6.2 MySQL数据库查询（范一迪）

从业务逻辑来看，数据库作为中转位置，要同时实现与前后端的交互功能。



为了实现前端对数据库的接口化访问，首先在本地MySQL进行查询语句测试，测试无误后将数据转存到云端MySQL数据库，也就是虚拟机上的MySQL。然后通过远程连接虚拟机上的MySQL数据库访问接口，实现对云端数据的修改与查询。



首先在虚拟机安装图形化界面并通过，可以更加方便的实现对云端数据库的操作，通过VNC Views客户端即可实现远程桌面控制。



当用户在前端界面进行相应查询请求后，前端将会返回对应的查询词段，通过封装好的API调用即可获得数据库相应的查询结果。

下面阐述如何通过mysql和python的交互，实现数据库查询的具体流程：

在获取NLP处理得到的用户查询需求后，首先通过IP连接到服务器以root用户访问MySQL。

将获取的查询需求拆分，通过SQL语言分条进行查询

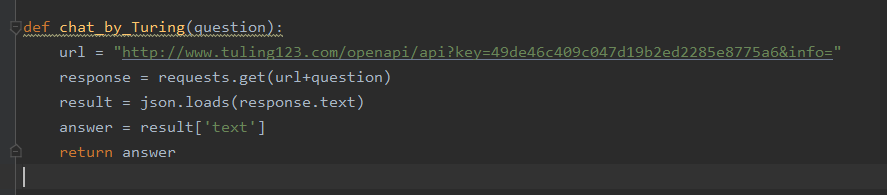
具体关键代码如下：



## 七、后端（曾旭祥、张凡）

### 7.1 聊天机器人

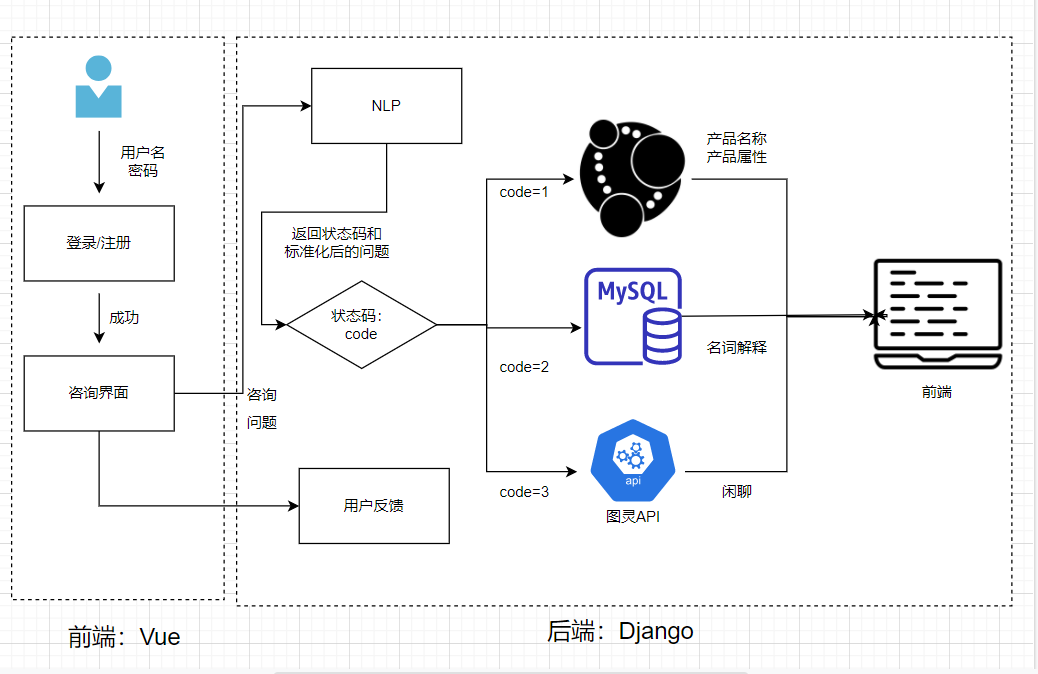
当用户提出的问题，在我们的系统中没有匹配到时，我们会自动调用我们的聊天机器人，和用户进行一个简单的沟通，让整个对话不那么尴尬，增强用户的体验。



我们的聊天机器人是通过调用图灵API的接口来实现的。在问题没有被匹配后，我们会将问题转发给图灵API，然后得到图灵API返回的结果，在返回给用户，从而达到一个和用户聊天的结果

### 7.2 后端与数据库部分

后端与数据库交互的部分主要分为三个模块，分别是名词解释、产品属性、以及聊天机器人。



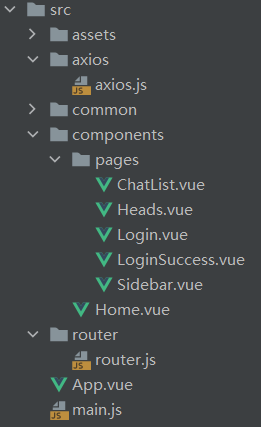
通过分词，我们将用户的问题分词为一下两种格式，第一种是{"列名1xxx":"属性","列名2xxx":"属性"}，第二种是{"属性值1xxx":"列名1","属性值2xxx":"列名1"}，其中的列名都是我们一开始构建好的一个词典。根据这个结果就可以去neo4j数据库中查找产品属性了，如果没有这样的结果，则去我们的词典中匹配其他的结果，匹配到结果后则去MySQL数据库中查找名词解释。如果词典中也没有结果，则判定这个问题是一个其他内容，并将这个问题当做聊天内容返回给图灵API，然后与用户进行一个聊天。

## 八、前端（杨继超）

前端采用vue框架及Element ui组件库进行设计。

1. Vue是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。
2. Element ui是一套为开发者、设计师和产品经理准备的基于Vue的组件库，提供了配套设计资源，帮助网站快速成型。由饿了么公司前端团队开源。

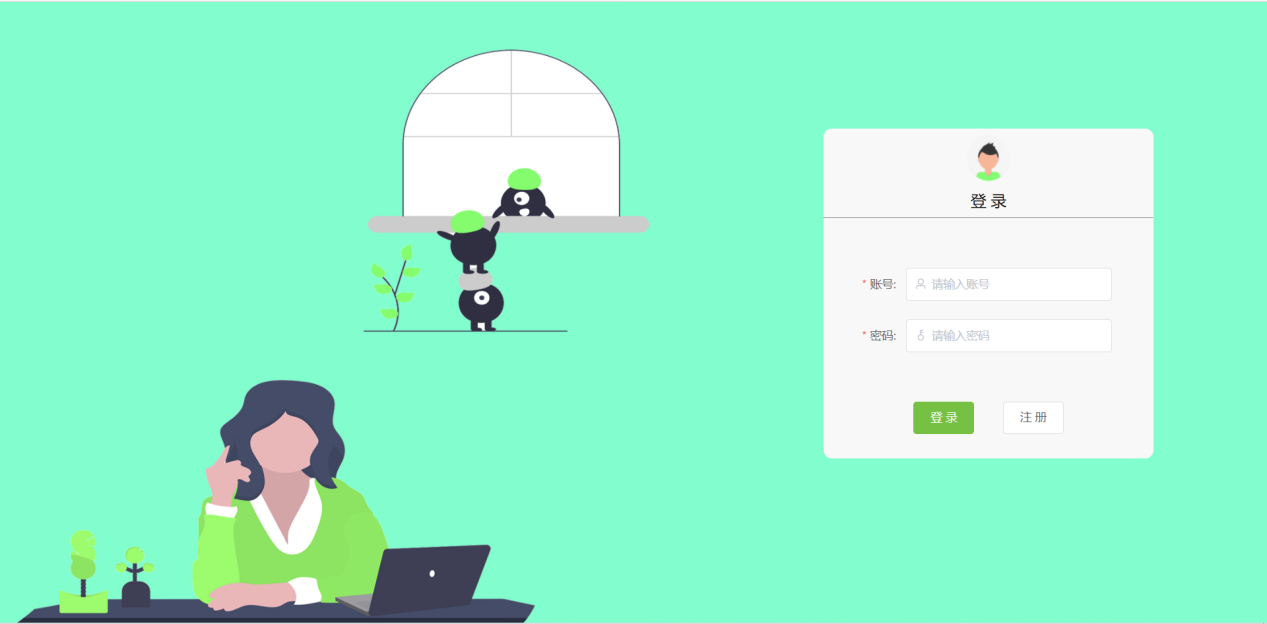
前端项目结构如下图所示：



前端结构大概分为三个部分：

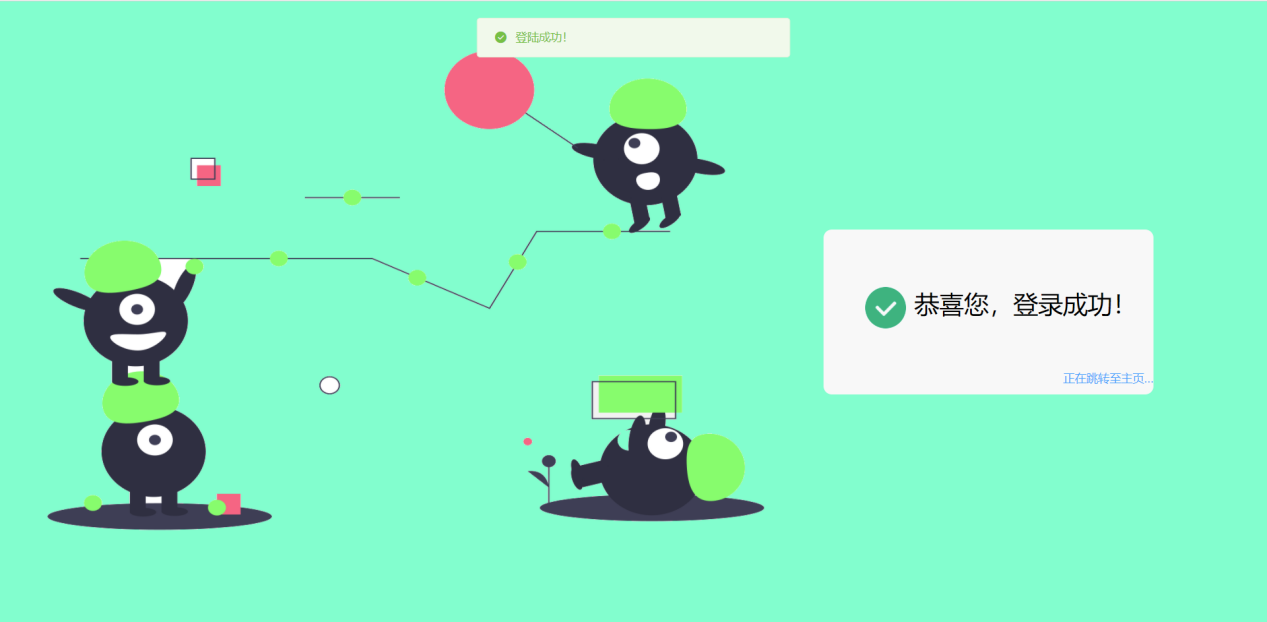
1. 页面部分，也就是components文件夹，里面存放前端页面设计的代码，共同组成了前端所有的页面。
2. 路由部分，也就是router文件夹，负责上面提到的所有页面间的动态切换。
3. 前后端交互部分，负责前后端的数据交互。在axios.js文件中封装了各个会用到的与后端交互传递接受数据的方法，方便在各个页面中进行调用。此外也用到了websocket进行数据交互，由于仅在聊天时会用到，websocket直接在聊天页面被加载时才加载初始化。

登录页面的设计如下图所示：

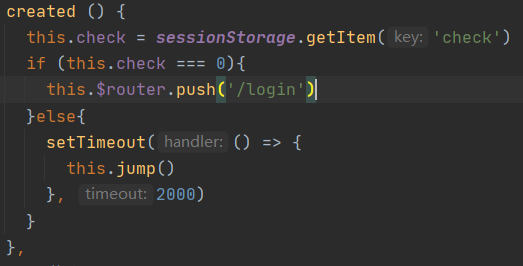


整个登录页面分为一个背景图片以及一个登录框。每个部件都是通过css样式的设计设置其大小并限制其位置，使其呈现图中的效果。

登录框主要由表单组件构成，并设置其填写规则，使其不能为空，用户点击登陆后通过axios将表单信息发送至后端并由后端查找数据库验证其是否匹配，返回一个布尔值到前端，若为true则登陆成功，跳转到登陆成功页面。

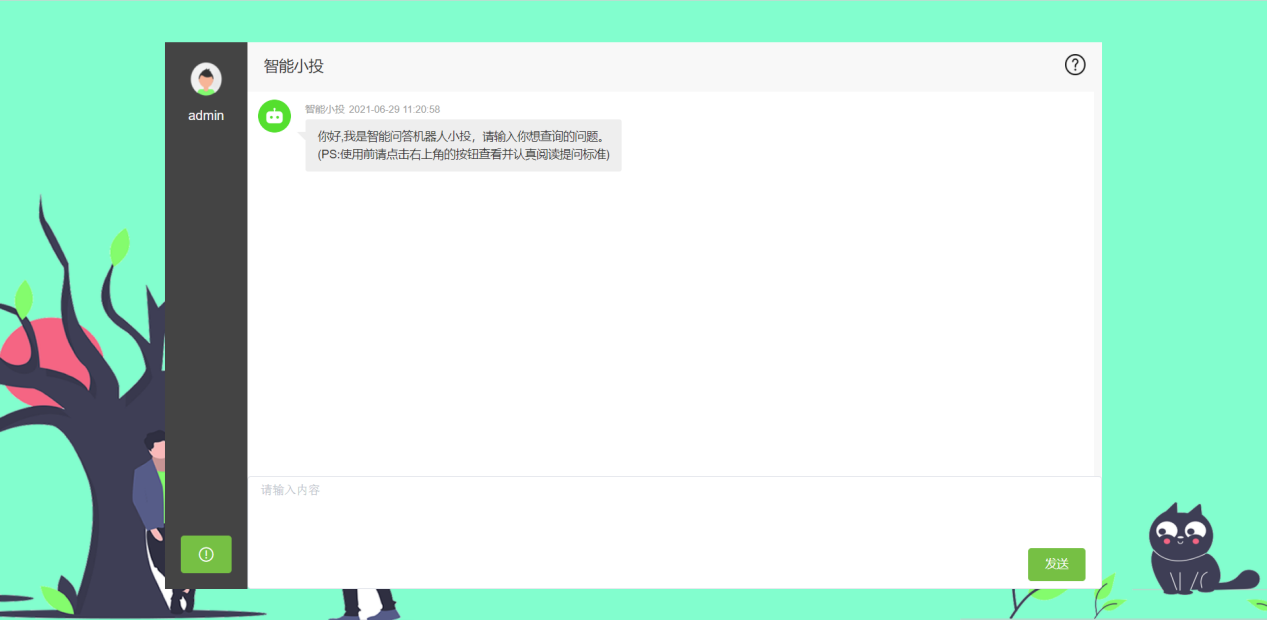


整个登录成功页面分为一个背景图片以及一个登录成功的提示框，并通过css样式的设计设置其大小并限制其位置，使其呈现图中的效果。



在跳转到登陆成功页面之后，会启动一个定时器，在两秒后自动跳转至对话页面即主页，用户也可以点击蓝色的“正在跳转至主页...”链接直接跳转至聊天页面。

对话页面的设计如下图所示：



整个对话页面分为三个页面组件组合而成：侧边栏、标题行、以及聊天框。每个部件都是通过css样式的设计设置其大小并限制其位置，使其呈现图中的效果。在加载页面时初始化websocket并持续监听，以便随时发送接收信息。

标题栏的组件有机器人的名字以及一个问号状的按钮，点击按钮会弹出给用户看的提问标准，方便用户问出正确格式的问题，以防得不到应有的正确回答。

侧边栏的组件有用户的头像、姓名以及一个反馈按钮，用户可以通过反馈按钮提一些改进意见与建议，这些建议通过axios传到后端接收并保存在数据库的反馈表中。

聊天框分为聊天消息的展示框以及输入框，输入框的组件有一个文本域用于输入内容以及一个发送按钮，用户点击发送按钮后调用sendMsg方法，在展示框上显示此条发送信息，并将消息框中的内容通过websocket发送至djiango交由后端处理，若为空则提示用户为输入信息。



在展示框部分主要是用v-for语句遍历records即聊天记录数组，只要有新的记录添加便在页面中进行显示，用v-if语句先判断此条记录的类型，若是用户发送的信息则显示在右侧，若是后端传回的信息则显示在左侧。

1. <http://www.cmbchina.com/cfweb/personal/default.aspx> [↑](#footnote-ref-1)
2. 因为部分理财产品只有说明书而没有风险揭示书，所以二者的数量存在差异 [↑](#footnote-ref-2)
3. 所有PDF均有的两个表格——理财计划要素表和投资比例表 [↑](#footnote-ref-3)
4. <https://blog.csdn.net/qiqi123i/article/details/90022799> [↑](#footnote-ref-4)
5. <https://github.com/HIT-SCIR/ltp> [↑](#footnote-ref-5)
6. <https://gitee.com/mirrors/pkuseg/> [↑](#footnote-ref-6)
7. <https://github.com/hankcs/HanLP> [↑](#footnote-ref-7)