**Lab2实验报告**

1. 思路概述

我们小组以Eclipse官网提供的bug缺陷报告数据作为原材料，将这些bug条目视为各种需求，进行分析。

首先利用爬虫技术，从网站上获取了10000份bug数据作为分析的源数据。

其次利用python的pandas包对获取的数据进行清洗和规范化处理，再利用自然语言处理库nltk对bug的文本进行分词处理，统计高频名词/动词。

最后利用通过筛选Comp属性以及Abstract属性中的关键词，对每个关键词进行评分，反过来利用这些关键词对每一条bug/需求进行综合评分，根据最终评分对所有bug进行等级评定，将模型评定结果与官方评定结果进行对比以完成模型检验。

1. 数据爬取
   1. 数据来源：<https://bugs.eclipse.org/bugs/>
   2. 网站分析：

要针对该网站进行大规模数据获取，首先要进行该网站页面源码的分析。该网站已经提供了十分规范的表格式数据，其中我们较为关注的属性是bug报告的ID，Product（所属产品），Component（所属组件/关键词），Summary（总结/摘要）

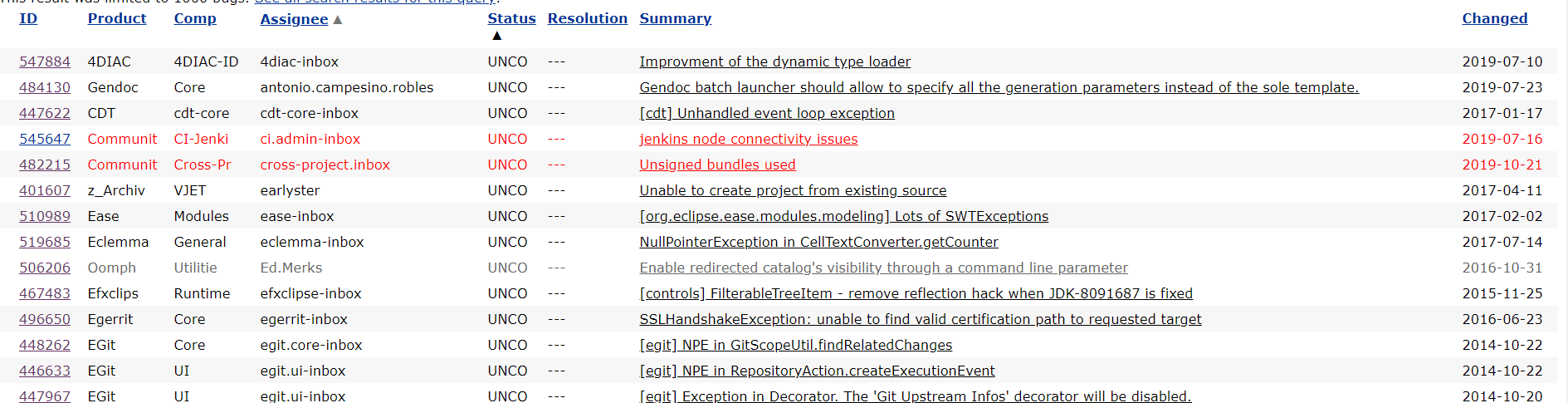


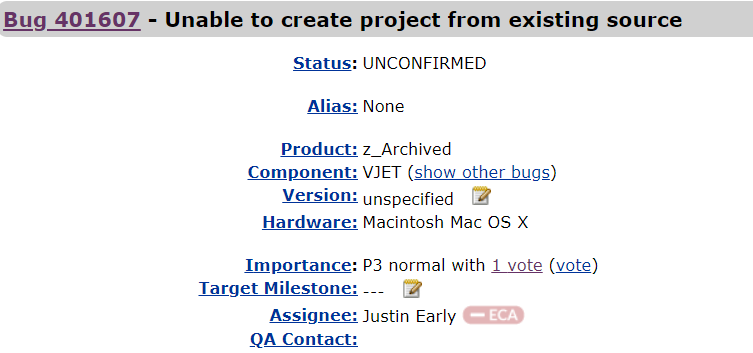
图1.网站整体概览

规范整齐的数据对于爬虫是十分友好的，再随便进入一个bug报告的链接，发现链接网址包含了一个bug条目的详细信息，以及官方给予这个bug的定级（Importance),这个定级将会作为我们实验的重要参考。注意到每个bug链接都拥有着相同的url结构”https://bugs.eclipse.org/bugs/show\_bug.cgi?id=???”，利用这个相同结构的特点，我们可以生成ID循环申请访问每一个bug报告对应的网站

}]BUC[8KIICJPA36KF3D91I

PD[F3W)KUH(KV_9V%V(7KLS

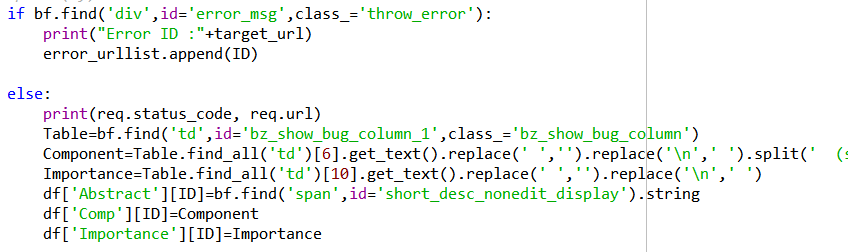
图2. 一致的网址结构

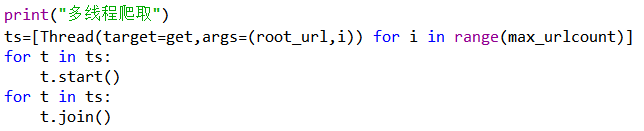


*图3 网站包含的详细bug信息*

* 1. 爬虫设计步骤（详见spyder2.0.py)：

1. 由于Eclipse的页面源码是静态源码，因此利用构造url并且requests.get(url)，很容易得到了一个页面的所有内容
2. 利用BeautifulSoup库定位到需要得到的bug信息（ID,Component,Importance...),简单处理一下得到文本信息
3. 因为初定的源数据数量为10000个bug，因此需要对10000多个网址进行访问获取（因为很多ID对应的bug信息已被删除或修复导致网址内容已经为空），所以采用并发编程的思想，创建了大约200个处理线程，多线程并发访问多个网址，最后大约5分钟把10000多个ID对应的网址数据获取完毕





*图4 BeautifulSoup处理和多线程访问网址*

* 1. 爬取结果（详见bugs.xlsx)：

获取的近万条数据中，Comp关键词和Abstract文本将会是我们后续数据处理和评分分析的重要属性，部分内容如下图所示

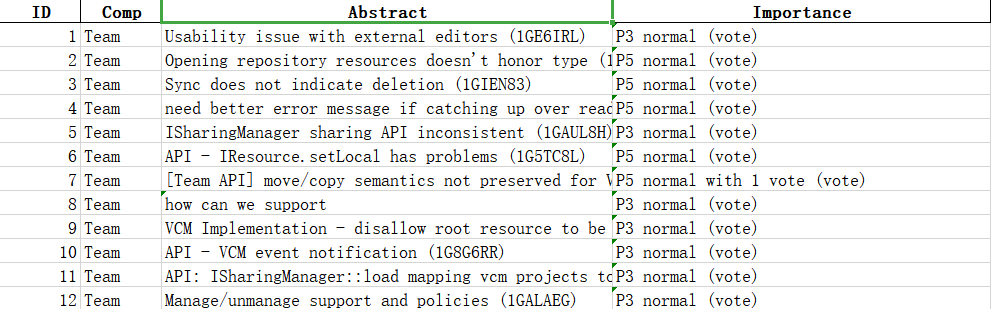


图5.数据结果概览

1. 数据处理

3.1目的：在后续的分析中仅对关键词进行分类/排序是很粗糙的，因此对于bug的Abstract文本内容分析比较重要，我们的想法是将文本中的名词和动词提取出来并加以统计分析，因为这两种单词（比如File,Compile,Update..)能反映一些bug信息的普遍内容

3.2 过程：利用python的ntlk库能快捷地完成词性标识这一步，但如果要进行词频统计，还需要进行单词的词形还原和大小写统一处理，比如单复数的file/files蕴含的意思相同，而动词的时态不同build/built也是相近的意思，必须加以处理和统一。除此之外，还要进行字母的小写转换和统一，比如File/file表示的是同一个单词。其余细节不再冗余详述，详见word\_segementation.py和word\_statistics.py

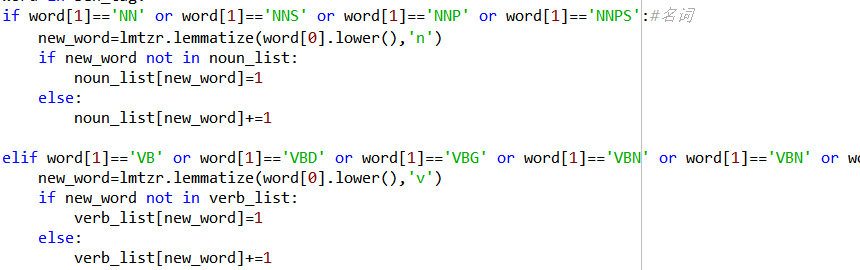


图6.利用词性标签进行统计计数

3.3 处理结果：根据P1~P5得到了每一级别的高频词统计结果（详见Px\_noun.xlsx和Px\_verb.xlsx)，统计之后发现各级别出现的高频词其实是有一定重合的，因此再进行了二次处理，统计了每个高频词在各级别出现的频率和权重，用作后续评分依据

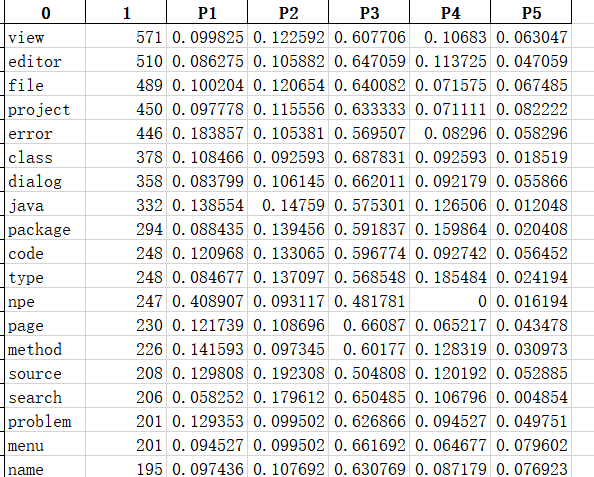
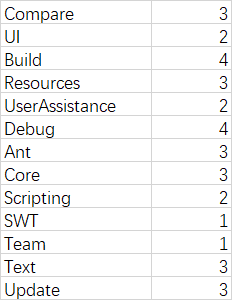


图7.部分高频词出现总数&&各级别出现频率

1. 数据分析与需求评分
   1. 抽取可用与评分的关键字

* 抽取Comp属性中可用于评分的关键词

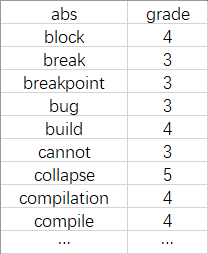
经过筛选，得出如下13个关键词，并对这13个关键词进行评分。



*Comp属性中筛选出来的关键词及其对应评分*

* 抽取Abstract属性中可用于评分的关键词

经过筛选，得出47个关键词，并对这47个关键词进行评分。



*Abstract属性中筛选出来的关键词及其对应评分*

* 根据抽取出来的关键词对原bug条目(bugs.csv)进行评分

编写代码，对Bug条目进行评分。

若Comp/Abstract属性中出现所抽取的关键词，则对该条目加上所出现关键词的分数

* 将评分后的条目进行分级

若分数grade≥8则该Bug为P1等级

若分数8＞grade≥6则该Bug为P2等级

若分数6＞grade≥2则该Bug为P3等级

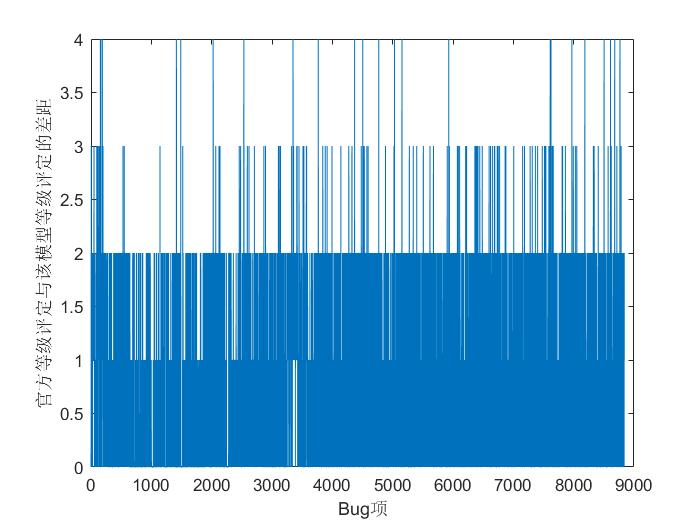
若分数2＞grade≥1则该Bug为P4等级

若分数1＞grade≥0则该Bug为P5等级

1. 排序结果检验

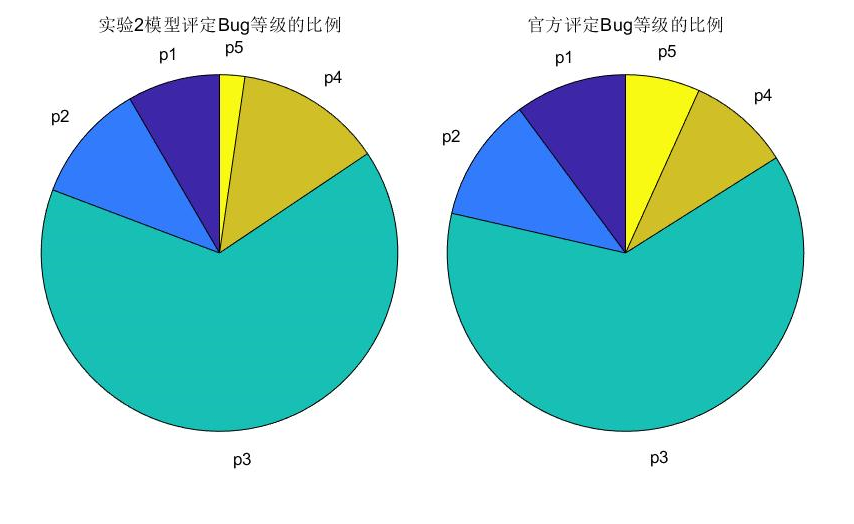
通过将该模型得出的等级结果与Bug原等级结果进行对比，得出如下结论：

* 8139条数据中有4158条数据的等级评定结果与原等级评定完全一致，一致率达到47.03%
* 下图为官方等级评定与本次实验模型评定结果的差距，可以看出，等级评分差距大概在1~2之间，模型相对可靠。



* 下图为官方等级评定以及本次实验模型等级评定后的等级分布比例

可以看出本次模型评定结果与官方评定结果在P1~P3部分比例大致相同，在P4、P5部分比例出现一定程度上的偏差。总体模型评定效果良好。



1. 误差分析

我们选择的bug分级方法是通过统计关键词在bug描述和类别中出现的次数和比例，通过人工方式筛选出部分有代表性的关键词作为评分标准，这种评分方法不一定能够客观反映bug的重要程度。由于关键词的筛选是在高频词中人工筛选，这样选出的关键词虽然有一定的代表性，但也有可能有遗漏，导致有些关键词未被正确选入或是选入了没有代表性的关键词。

在一开始的数据统计过程中，我们发现P3等级的bug占了bug总数的一半以上，而其他等级的bug数量少，因此我们认定大多数重要性不太高的关键词的等级应该被评为3，重要的关键词等级为5，而不重要的关键词等级为1，这样的评级方式受到主观判断的影响很大，导致关键词的重要性有可能受到误判。

在统计每一条bug的具体评分时，我们采取了每当出现一个关键词时，直接将其评分累加的方式。这样的方式有可能导致当一些关键词之间的出现有一定程度的逻辑关联时，使得该条bug的统计得分虚高。

本实验所采用的的关键词评分方式的默认条件是：每一个关键词有着不变的评分，每一条bug中出现的关键词越多，该条bug评分越高。但是实际情况并非如此，同样一个关键词在不同的bug中可能体现出不同的评级，统计数据只能一定程度反映这个关键词的评级倾向，并不能决定具体某一条bug的评级。而多个关键词同时出现的bug也不一定评分高，有些关键词处于相近的等级，那么他们组合后的等级可能也是相近的，简单相加的方式无法反映这样的可能。

总而言之，我们采取的实验方法只是一种粗糙的、大致模拟的分级方式，默认的条件与现实情况有着较大差异，因此得出的实验结果与原本结果有较大差异是正常情况。

注：小组成员及成绩比例

|  |  |  |
| --- | --- | --- |
| 学号 | 姓名 | 成绩比例 |
| 171860657 | 徐浩 | 0.25 |
| 171860677 | 吴鸿祜 | 0.25 |
| 171868570 | 周吴成 | 0.25 |
| 171870642 | 王子豪 | 0.25 |