软件需求工程-实验报告

Lab2：软件需求的优先级排序

2019.11

目录

目录

一、 小组成员

学号	姓名	成绩分配比例
161220056	敬舒舒	0。25
161271030	张梦窈	0。25
161290019	吴雨昕	0。25
171860519	李培剴	0。25

二、 数据获取

本次实验基于eclipse项目的缺陷报告数据，将这些数据当作软件需求，对这些需求进行优先级排序。通过爬虫获取到约55万条需求信息。

2.1 获取网页数据

利用python进行爬虫，我们需要遍历所有缺陷报告页面来获取信息。目标网页的url格式较为简单，每个网页的格式为“bug展示地址+bug_id”，比如第100个缺陷报告的页面为“https://bugs.eclipse.org/bugs/show_bug.cgi?id=100”。

需要注意的是，虽然缺陷报告的id基本都是从小到大逐次加一，但存在id无效的情况，因此在爬虫过程中需要判断该id是否有效。判断方法为检查网页数据中是否有无效提醒信息'You must enter a valid bug number!'。

同时由于数据条数过大，中途很可能出现超时、与主机断开连接等错误并退出，这里将请求超时时间设置为5s，若是超时或发生异常则重新发送请求直到成功为止。

2.2 获取需求

通过爬虫得到缺陷报告网页后，我们需要从该文本中提取出各种相关数据，这里使用了bs4中的BeautifulSoup库，利用该库的find_all、findNext方法定位到数据位置，获得id、title、status、product、component、importance等19类信息。同时对数据进行一定程度的预处理，包括统一格式、去除无关信息。

三、 需求排序

3.1 数据预处理

预处理指将数据条目中需要的属性筛选出来，以及对title中的关键字进行提取。

读写csv文件时用csv库中的reader和writer函数。用reader读时，返回值是文件中每行的列表，每一行都是由各属性下的字符串组成的列表。在属性中保留了id, title, status, importance四项，除title外的三项属性直接保留，对title进行处理。首先将title中的字母全部改为小写，然后对照nltk中的英语停词表跳过停止词。若不是停止词，再用lemmatize函数进行词型还原，并保留英文单词。经过上述处理得到较为理想的标题关键词。随后将原属于同一数据条目的属性组成一个列表也即一行，再将各行再组成一个列表，于是便可以用writerows函数将其直接写出到新的csv文件。

3.2 正式排序

我们对需求排序的方法是等级排序，因为这里直接将每个等级视作一种类别，五种等级视作五个种类，然后用多分类的方法进行等级评定。实验中训练集基本是前50万条数据，测试集是50万~51万的一万条数据。由于某些模型如决策树和随机森林运算比较复杂耗时过长，这两种模型采用的不是前50万条，是30万到50万的20万条数据和40万到50万的

10万条数据。

3.2.1 Tf-idf词转词向量

在3.1预处理得到实际关键词后，进一步使用tf-idf方法将关键词转为词向量。tf-idf（Term Frequency-Inverse Document Frequency，词频-逆文件频率）是一种有效的提取文本库中某一段文本的特征的有效方法。tf指的是一个词语在局部文本中出现的次数，通常还会被归一化以防它偏向长的文件；idf指的是如果包含某个词语的文档越少，idf就越大。tf-idf综合考虑了在文件中出现的次数和和在整个文本库中出现的次数。一个词的tf-idf值越大，在局部文本中出现的次数越多，在全局中出现的次数越少，说明它越能够作为这个局部文本的特征来进行归类。

这里使用的api是sklearn.feature_extraction。text里的TfidfVectorizer。

3.2.2 朴素贝叶斯

朴素贝叶斯是实验用到的所有算法中复杂度最低的一种，它实现简单，没有迭代，学习效率高，在大样本量下会有较好的表现。而且它的强假设——所有特征条件相互独立在本次实验中也得到了满足。

算法的基本思路如下：

首先运用贝叶斯定理求解后验概率中最大者：

$$P(C_i \mid X) > P(C_j \mid X),\ 1 \le \mathrm{j} \le \mathrm{m},\ \mathrm{j} \ne \mathrm{i}$$

然后将其对应的的类别作为预测类别。至于其求法，可根据贝叶斯定理得到以下公式

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)}$$

由于我们假设各个特征条件相互独立，所以特征向量的后验概率可以通过每个特征值的后验概率的乘积得到。

$$P(X \mid C_i) = \prod_{k=1}^{n} P(X_k \mid C_i)$$

本次实验中，我们可以认为词与词之间是离散的，可以直接用类中符合属性的元组数除以类的元组总数进行概率的计算。

$$P(x_k \mid C_i) = \frac{S_{ik}}{S_i}$$

这里使用的api是sklearn.naive_bayes里的MultinomialNB。

3.2.3 决策树

决策树基于特征属性进行分类，决策树归纳是一种从有类标号的训练元组中学习决策树的算法，其优点包括：模型具有可读性，计算量小，分类速度快。决策树算法有ID3，C4.5，CART等。

决策树的构造过程一般分为3个部分，分别是特征选择，决策树生产和决策树裁剪。

① 特征选择。 特征选择表示从众多的特征中选择一个特征作为当前节点分裂的标准，如何选择特征有不同的量化评估方法，从而衍生出不同的决策树，其中C4。5通过信息增益比选择特征。

② 决策树的生成。 根据选择的特征评估标准，从上至下递归地生成子节点，直到数据集不可分则停止决策树停止生长。这个过程实际上就是使用满足划分准则的特征不断的将数据集划分成纯度更高，不确定行更小的子集的过程。对于当前数据集的每一次划分，都希望根据某个特征划分之后的各个子集的纯度更高，不确定性更小。

③ 决策树的裁剪。 决策树容易过拟合，一般需要剪枝来缩小树结构规模，缓解过拟合。

ID3算法的核心是根据信息增益最大的准则，递归地构造决策树;算法流程如下:

1) 如果节点满足停止分裂条件(所有记录属同一类别 or 最大信息增益小于阈值)，将其置为叶子节点
2) 选择信息增益最大的特征进行分裂;
3) 重复步骤1-2，直至分类完成。

这里使用的api是sklearn.tree中的DecisionTreeClassifier，对应算法是ID3。

3.2.4 随机森林

随机森林就是通过集成学习的思想将多棵树集成的一种算法，它的基本单元是决策树，而它的本质属于机器学习的一大分支——集成学习(Ensemble Learning)方法。随机森林的名称中有两个关键词，一个是“随机”，一个就是“森林”。从直观角度来解释，每棵决策树都是一个分类器，那么对于一个输入样本，N棵树会有N个分类结果。而随机森林集成了所有的分类投票结果，将投票次数最多的类别指定为最终的输出，这就是一种最简单的 应用了Bagging方法的集成学习。

这里使用的api是sklearn.ensemble中的RandomForestClassifier。

3.2.5 支持向量机

SVM(Support Vector Machine，支持向量机)是一种对线性和非线性数据进行分类的方法。SVM使用非线性映射把原训练数据映射到较高的维上。在新的维上，它搜索最佳分离超平面。使用到足够高维上的，合适的非线性映射，两个类的数据总可以被超平面分开。可以画出无限多条分离直线。我们想找出“最好的”一条，即在先前未见到的元组上具有最小分类误差的那一条。扩展到n维，我们希望找出的即为超平面。SVM寻找最佳超平面，具体地说就是最大边缘超平面。

这里使用的api是sklearn.svm中的LinearSVC。

3.2.6 KNN

k最近邻方法是一种惰性学习法，也称“基于实例的学习法”。它基于类比学习，即通过将给定的检验元组与和它相似的训练元组进行比较来学习。训练元组用n个属性描述，每个元组代表n维空间的一个点，所有的训练元组都存放在n维模式空间中。当给定一个未知元组时，kNN算法搜索模式空间找出最接近其的k个训练元组，并将其指派到k个最近邻中的多数类。

这里使用的api是sklearn.neighbors中的KNeighborsClassifier。

四、 效果分析

对于需求优先度的排序，由于分为了P1~P5五个等级，对应为多分类问题。因此主要评价了预测模型的四项指标：**精度**，**查准率**，**查全率**和**f1 score**。

精度主要用来衡量分类正确的样本数占样本总数的比例。

查准率是指：预测为正的样例中有多少为真正的正样例。对于本多分类问题，实际上可以求每个分类的查准率，取加权平均或绝对平均。对应下方报告未weighted avg和macro avg。

查全率是指原来样本中的正例有多少被预测正确，类似查准率，我们依然对每个分类取平均值。

f1度量是查准率和查全率的一个调和平均，公式为：

$$\frac{1}{F1}=\frac{1}{2}(\frac{1}{P}+\frac{1}{R})$$

五种预测模型的整体报告如下图所示：

bayes report:

	precision	recall	f1-score	support
1	0.00	0.00	0.00	84
2	0.00	0.00	0.00	308
3	0.95	1.00	0.98	9538
4	0.00	0.00	0.00	47
5	0.00	0.00	0.00	23
accuracy			0.95	10000
macro avg	0.19	0.20	0.20	10000
weighted avg	0.91	0.95	0.93	10000

decisionT report:

	precision	recall	f1-score	support
1	0.00	0.00	0.00	84
2	0.06	0.03	0.04	308
3	0.95	0.98	0.97	9538
4	0.00	0.00	0.00	47
5	0.00	0.00	0.00	23
accuracy			0.93	10000
macro avg	0.20	0.20	0.20	10000
weighted avg	0.91	0.93	0.92	10000

randomF report:

	precision	recall	f1-score	support
1	0.00	0.00	0.00	84
2	0.13	0.01	0.02	308
3	0.95	1.00	0.97	9538
4	0.00	0.00	0.00	47
5	0.00	0.00	0.00	23
accuracy			0.95	10000
macro avg	0.22	0.20	0.20	10000
weighted avg	0.91	0.95	0.93	10000

knn report:

	precision	recall	f1-score	support
1	0.07	0.04	0.05	84
2	0.33	0.00	0.01	308
3	0.95	1.00	0.97	9538
4	0.00	0.00	0.00	47
5	0.00	0.00	0.00	23
accuracy			0.95	10000
macro avg	0.27	0.21	0.21	10000
weighted avg	0.92	0.95	0.93	10000

svm report:

	precision	recall	f1-score	support
1	0.00	0.00	0.00	84
2	0.20	0.01	0.01	308
3	0.95	1.00	0.98	9538
4	0.00	0.00	0.00	47
5	0.00	0.00	0.00	23
accuracy			0.95	10000
macro avg	0.23	0.20	0.20	10000
weighted avg	0.92	0.95	0.93	10000

整体来看，五种预测模型的准确度较为接近，其中仅决策树模型准确度略低

（0.02），而查准率同样接近。比较大的不同之处，如果采用绝对平均方式计算查准率，则knn模型的平均查准率明显较高。初步分析是由于样本中对于P1 P2等高优先级需求的样本总数较少，因此采用加权平均时权重较低。而knn模型对于数量比较少的高优先级需求的预测较为准确，如P2的查准率达到33%。 在采用加权平均时，该优势并不明显，而采用绝对平均时，该优势得以体现。

预测模型的四项指标详细对比如图：

```
bayes accuracy: 0.9537
decisionT accuracy: 0.9314
randomF accuracy: 0.9495
knn accuracy: 0.9501
svm accuracy: 0.9522

bayes precision: 0.19075907590759075
decisionT precision: 0.20378931413414172
randomF precision: 0.21693846954073984
knn precision: 0.27147626436351857
svm precision: 0.23083892953793725

bayes recall: 0.19997903124344726
decisionT recall: 0.20158681746016618
randomF recall: 0.2009834891466261
knn recall: 0.2069324887735456
svm recall: 0.20092126368075203

bayes f1 score: 0.19526027537492963
decisionT f1 score: 0.20162971361534252
randomF f1 score: 0.19848296039947064
knn f1 score: 0.2056452028953677
svm f1 score: 0.19764767305265904
```

同时为了衡量预测模型预测结果与实际结果的偏离程度，我们计算了预测结果与实际结果的Hamming Loss （平均汉明距离） 如图所示：

```
bayes hamming loss: 0.0463
decisionT hamming loss: 0.0686
randomF hamming loss: 0.0505
knn hamming loss: 0.0499
svm hamming loss: 0.0478
```

其中决策树模型的Hamming Loss较高，表明其预测结果的偏离程度较高，而其余四种模型偏离程度较为接近。