**实验报告书**

|  |  |
| --- | --- |
| 实验课程名称 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| 指导教师姓名 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| 学生姓名 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**一、实验目的、意义**

本实验旨在通过资料查阅和上机实验，使学生加深了解数据分析与建模的理论和方法，实现一套从数据处理、数据分析、数据建模和评估的完整数据挖掘算法流程。

**二、实验基本原理与方法**

Python数据处理的基本操作、分类模型算法的选择、sklearn库的使用。数据分析的基本概念。

**三、实验内容及要求**

（1）实验内容：

医疗对话意图分类，基于CMID数据集，为医疗对话系统中的对话意图分类构建一个模型，评价所构建模型的性能，给出模型性能的混淆矩阵。分类要求对label\_4class进行分类。

（2）数据集介绍：CMID.json

CMID.json是一个关于医疗对话意图判断的数据集，共有5个字段。例子如下

{

"originalText": "间质性肺炎的症状?",

"entities": [{"label\_type": "疾病和诊断", "start\_pos": 0, "end\_pos": 5}],

"seg\_result": ["间质性肺炎", "的", "症状", "?"],

"label\_4class": ["病症"],

"label\_36class": ["临床表现"]

}

其中，每个字段的含义如下：

"originalText" ：医疗对话系统原始输入文本数据；

"entities"：输入文本中包含的知名实体信息，包含知名实体的类别及起始位置信息；

"seg\_result"：输入文本分词后的结果

"label\_4class"：医疗对话意图类别

"label\_36class"：医疗对话意图二级分类信息

医疗对话意图分类详细分类信息：

**"label\_4class"：**

病症 药物 治疗方案 其他

**"label\_36class"：**

**病症**：定义，病因，临床表现，相关病症，治疗方法，推荐医院，预防，所属科室，禁忌，传染性，治愈率，严重性

**药物**：作用，适用症，价钱，药物禁忌，用法，副作用，成分

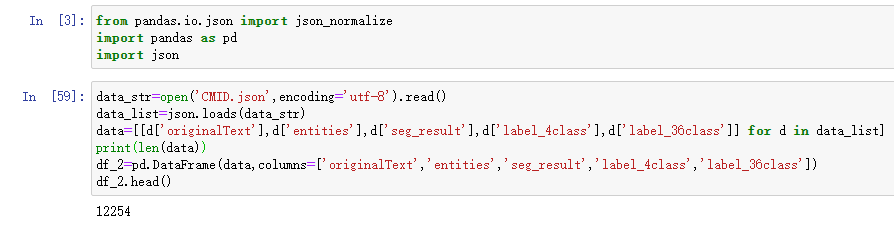
**治疗方案**：方法，费用，有效时间，临床意义/检查目的，治疗时间，疗效，恢复时间，正常指标，化验/体检方案，恢复

**其他**：设备用法，多问，养生，整容，两性，对比，无法确定

**四、实验过程**

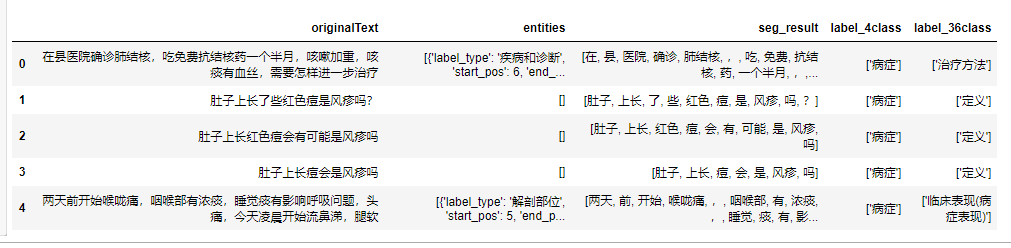
**（1）数据分析**

下载数据集，查看数据集内容，由于数据集是json格式，为了看得更加清晰，将数据集从json格式转换成DataFrame格式，方便查看：



将json转为DataFrame格式，可以看到，数据集总共有12254条

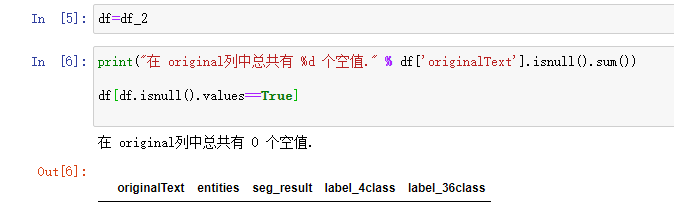
查看前5条数据详情



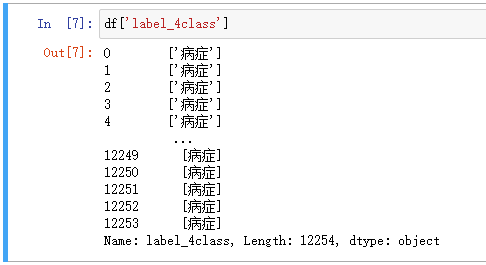
数据集总共有5列。分别为originalTet、entities、seg\_result、label\_4class和label\_36class.分析内容，是根据originalText:医疗对话系统原始输入文本数据，进行预测label\_4class：医疗对话意图类别。

**（2）数据处理**

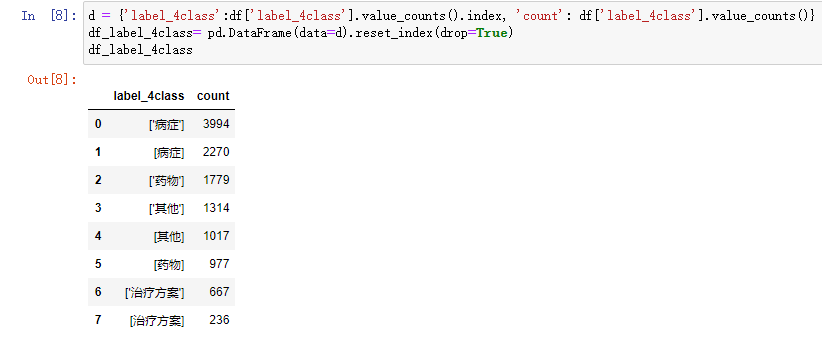
首选查看是否有空缺值



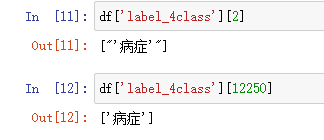
发现数据完整，没有空缺值。接着查看分类标签：

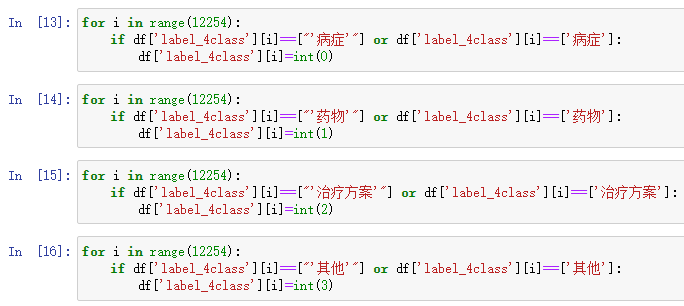


统计每个类别的数量

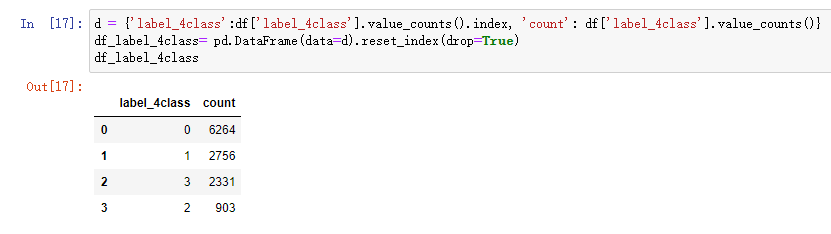


从上图可以发现，标签有重复的部分，比如[“病症”]和[“”病症””]等等，所以需要进行标签的合并处理。





将字符标签转化为数字类型，其中类别“病症”对应0，“药物”对应1，“治疗方案”对应2，“其他”对应3。接着统计每个类别的个数。

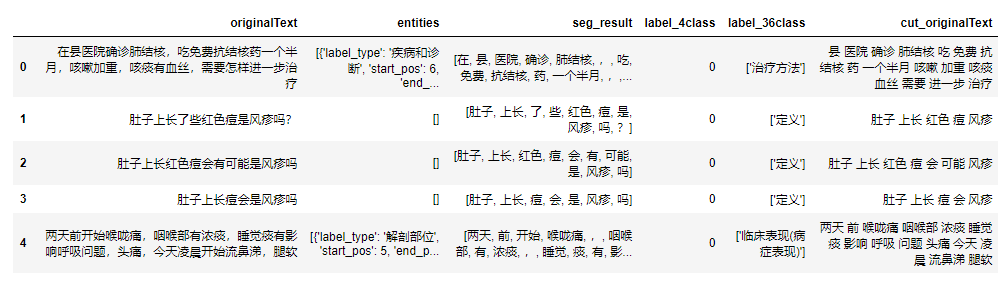


由于医疗原始文本内容都是中文,所以要对中文进行一些预处理工作,包括删除文本中的标点符号,特殊符号,还要删除一些无意义的常用词(stopword),因为这些词和符号对系统分析预测文本的内容没有任何帮助,反而会增加计算的复杂度和增加系统开销,所有在使用这些文本数据之前必须要将它们清理干净。



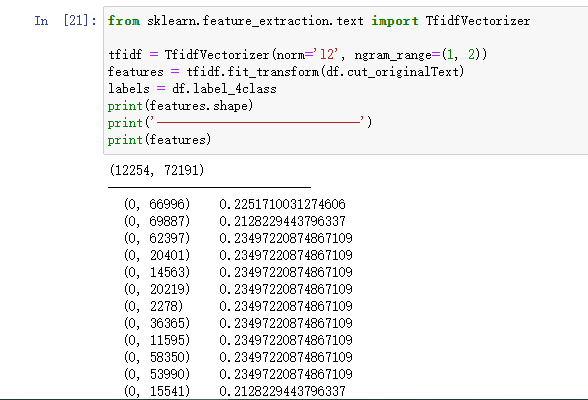
中文停用词包含了很多日常使用频率很高的常用词,如 吧，吗，呢，啥等一些感叹词等,这些高频常用词无法反应出文本的主要意思,所以要被过滤掉。

接着我们进行自己的分词



接下来计算cut\_originalText的 TF-IDF的特征值，TF-IDF（term frequency–inverse document frequency）是一种用于信息检索与数据挖掘的常用加权技术。TF意思是词频(Term Frequency)，IDF意思是逆文本频率指数(Inverse Document Frequency)。TF-IDF是在单词计数的基础上，降低了常用高频词的权重,增加罕见词的权重。因为罕见词更能表达文章的主题思想,比如在一篇文章中出现了“中国”和“卷积神经网络”两个词,那么后者将更能体现文章的主题思想,而前者是常见的高频词,它不能表达文章的主题思想。所以“卷积神经网络”的TF-IDF值要高于“中国”的TF-IDF值。

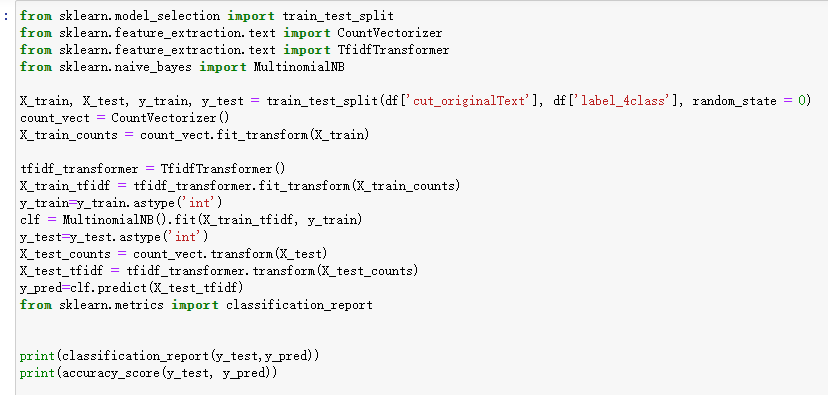
这里我们会使用sklearn.feature\_extraction.text.TfidfVectorizer方法来抽取文本的TF-IDF的特征值。这里我们使用了参数ngram\_range=(1,2),这表示我们除了抽取评论中的每个词语外,还要抽取每个词相邻的词并组成一个“词语对”,如: 词1，词2，词3，词4，(词1，词2)，(词2,词3)，(词3，词4)。这样就扩展了我们特征集的数量,有了丰富的特征集才有可能提高我们分类文本的准确度。参数norm='l2',是一种数据标准划处理的方式,可以将数据限制在一点的范围内比如说(-1,1)

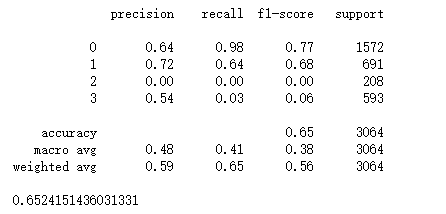


**（4）数据建模**

接下来是分类器的选择，我们首先尝试朴素贝叶斯。

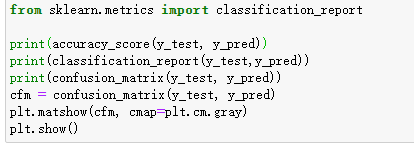
朴素贝叶斯分类器最适合用于基于词频的高维数据分类器，这里我们使用的是sklearn的朴素贝叶斯分类器MultinomialNB，我们首先将originlText转换成词频向量,然后将词频向量再转换成TF-IDF向量，这里我们还是按照一般的方式将生成TF-IDF向量分成两个步骤：1.生成词频向量. 2.生成TF-IDF向量。最后我们开始训练我们的MultinomialNB分类器。

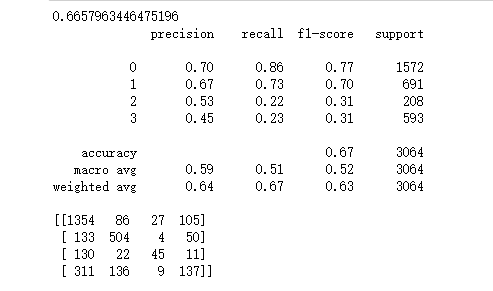


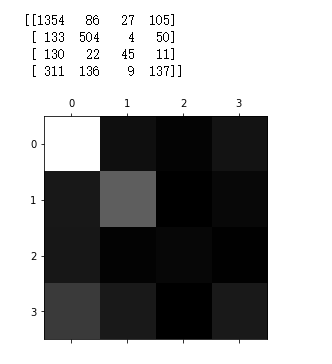


尝试了朴素贝叶斯、LinearSVC、随机森林和GBDT。最终选择GBDT作为我们的模型。

**（5）模型评估**







结果分析，最终分类准确率大约为0.666,根据我们模型的混淆矩阵，发现第0类也就是“病症”的结果很准确，第2类差错过大，我们分析，可能是因为数据处理中的词向量表示没有做好，使得第0 类和第2类的混淆度较高。另外，通过进一步地模型调参，效果会接近0.7。

1. **实验总结**

通过这次实验过程，让我熟悉了整个数据分析的过程，在实际中数据可能会有各种各样的问题，因此整我基本的python数据处理方法是必须的。

在数据建模过程中，对于模型的选择也是一个不断尝试的过程，通过模型不断尝试，调参，进而找到最佳模型和最佳参数，让我对分类模型有了一个更加深入的了解。