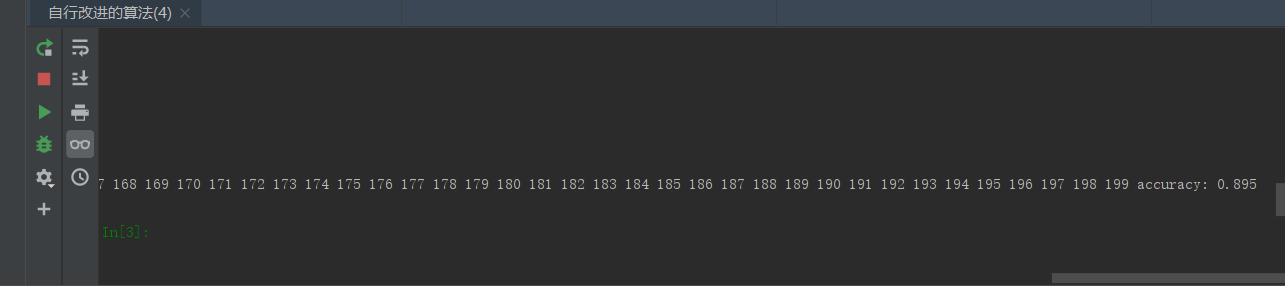
首先尝试使用朴素贝叶斯模型进行分类，见“生成bow.py”. 生成词带，每一列为一个词（经过去标点、转小写等操作后的原词），每一行为一条数据（共12W行），每一个的内容为这个词在这条新闻中出现的次数。之后通过p(c|d)=p(c)\*p(d|c)/p(d)的算法来计算测试集中的新闻属于4个类中哪一个的概率最大，以此来判断属于哪一类。

经过反复试验，这样大的数据量已经接近内存极限，频繁导致电脑死机，遂放弃。

于是开始尝试别的方法，见文件‘自己的算法.py’，由于导致内存不足的原因是储存词袋所需内存过大，只能从改变词袋开始。新方案是词袋仍然以单词为列，但是总共只有4行，每一行代表一个分类，期中每一格记录这个分类中这个单词出现的概率（p=这个单词的个数/总单词数）。

在这个模型下假设为伯努利概型，则新数据的p(c|d)可以这样表示：p = ∏｛C（ni，n总）\* Pi^ni\*(1-Pi)^(n总-ni)｝。考虑到数据较小，引入log来度量。

使用新模型的效果根据本人估计应该与贝叶斯等价。处理速度大约为一秒一条（还是不够快啊），测试使用前200条，正确率为0.895. 

改进方法：目前算法速度应该比朴素贝叶斯快。可以提升的地方比如：去除频率较低的词，只保留高频词；用c来做数据运算（或者调用基于C的库）；