## 通过数据挖掘验证《红楼梦》后四十回真伪

本文思路，（1），提取每一章关键词权重并向量化。（2）K均值聚类。

首先，词频统计的方法有失偏颇，故用TF-IDF方法计算关键词的权重。其次，分类的方法预设了标签，是一种有监督学习模式，聚类是无监督学习模式，是根据数据自身特性进行“自动站队”。私以为，区分《红楼梦》前后不同用聚类的方法更有直观且具有说服力。

参见：https://www.zhihu.com/question/42044303/answer/107836313

TF-IDF：一种[统计方法](http://baike.baidu.com/view/1379337.htm)，用以评估一字词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在[语料库](http://baike.baidu.com/view/686705.htm)中出现的频率成反比下降。（百科）

K-means算法：是很典型的基于距离的[聚类](http://baike.baidu.com/view/31801.htm)算法，采用距离作为相似性的评价指标，即认为两个对象的距离越近，其相似度就越大。该算法认为簇是由距离靠近的对象组成的，因此把得到紧凑且独立的簇作为最终目标。（百科）

运用工具：Python 2.7，Weka 3.8

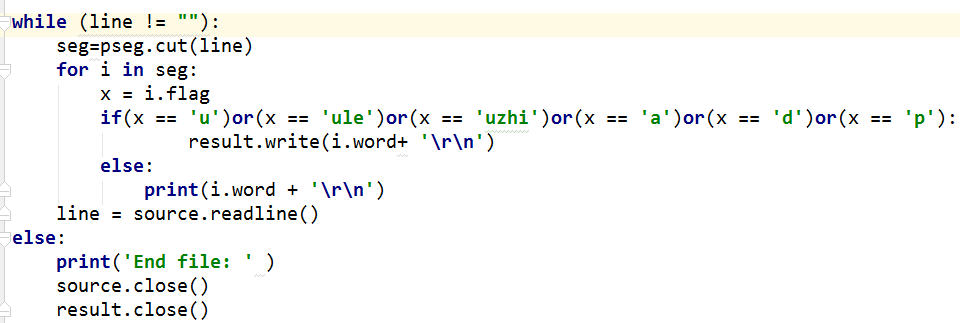
1 特征提取

1.1运用jieba进行分词，提取每一章中的助词，虚词，形容词和介词，并计算此类词在每一章中的TF\_IDF权重。Jieba分词是Python中文分词组件，可以对中文文本进行分词，关键词提取和词性标注。

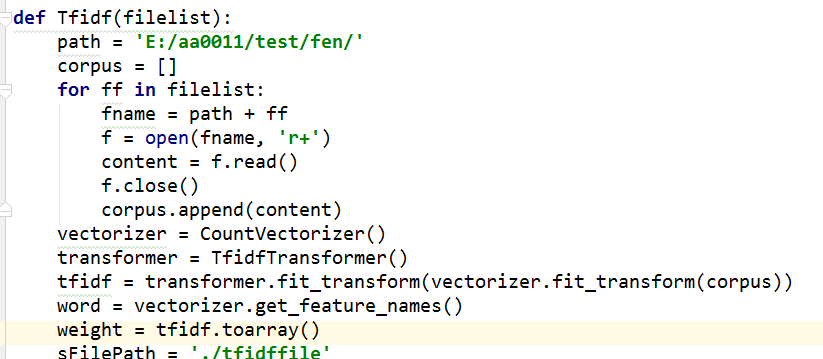
首先我想到的是，对全文进行分词，再提取每个词的TF-IDF权重。但这样做有两个缺点，一是数据太大，噪声太多。这样矢量化的文本有2万多维，且有许多垃圾数据，比如，“宝玉”和“宝”，“玉”划分成了三个词。二是有些名词，代词的出现的确影响结果。例如，黛玉，这个词，无论后四十回是不是续作，其出现的频率也应该是比较低的。

不同作者所用助词，虚词，形容词和介词的用法多不同， 在jieba分词中能对词性进行标注，故我提取了文章中所有的上述词型，并计算TF-IDF权重。得到1536个词，组成120\*1536的向量。

程序分两部分，第一部分提取文章中上述词型，组成词库。



第二部分计算每章中上述词的TF-IDF权重。



PS：上述两部分可写在一起……第二部分代码部分借鉴于http://www.tuicool.com/articles/feIji2

2 利用Weka进行K均值聚类

WEKA：作为一个公开的数据挖掘工作平台，集合了大量能承担数据挖掘任务的机器学习算法，包括对数据进行预处理，分类，回归、聚类、[关联规则](http://baike.baidu.com/view/1076817.htm)以及在新的交互式界面上的可视化。（百科）

Weka的打开格式为arff，将得到的向量处理为arff格式。再进行k均值聚类。操作过程为，载入数据，选择k均值聚类，聚类数设置为2类，选择合适的seed（随机种子），使均方差最小。下面是均方差较小时的聚类结果。

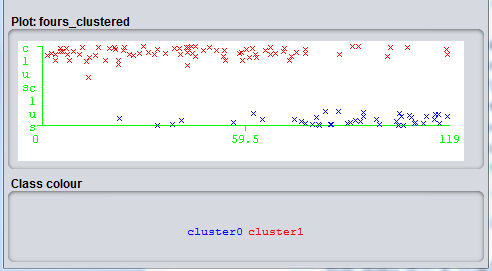


图1 Numclusters=2， Seed=38，Sum of squared errors=3483时的聚类结果

聚类占比为：

Clustered Instances

0 42 ( 35%)

1 78 ( 65%)

为了证明聚类结果的有效性，取聚类数为3时：

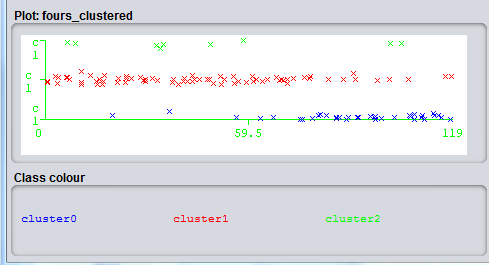


图2 Numclusters=3，Seed=38，Sum of squared errors=3448时的聚类结果

聚类占比为：

Clustered Instances

0 37 ( 31%)

1 74 ( 62%)

2 9 ( 8%)

3结果分析

上两图横轴代表回数，显而易见，前八十回和后四十回属于不同的类别，也就是《红楼梦》后四十回很有可能是续作。在聚类数为3时，仅有8%被归到第三类，这说明其余两类内部特征相似性比较强，分为两类是合适的。《红楼梦》后四十回基本可以断定为伪作。

4不足及改进

由于分词的不精确性，仅用抽取的部分特征代替全局。

没有对刘心武续写的进行分析……那样结果应该更明显。。