**实 验 报 告**

**课程名称** Python选修课

**实验项目** 《红楼梦》前八十回与后四十回的语言特点分析

**实验仪器**  个人计算机

**学 院** 计算机学院

**专 业** 计算机大类

**班级/学号** 计类1801/2018011140

**学生姓名** 张京宇

**实验日期** 2019.4月

**成 绩** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**指导教师** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

一 、实验目的

通过本实验，掌握python语言的基本数据类型列表、字典的使用和操作，以及文件的读写操作，并用python语言完成实际任务。

二、 实验内容 （20分）

《红楼梦》前八十回与后四十回的语言特点分析：掌握需求分析方法，学习第三方库jieba的安装和使用。用字符串、列表、字典操作完成任务。本实验分为五个阶段：预处理实验文本、编写实验代码初步实现用程序对词频进行统计、对词频数据进行多维度分析处理、初步得出结论、论证结论。

三、实验原理（20分）

首先对实验文本经行预处理，根据标点符号，标出每一句话，将段落转化句子。其次利用开源分词组件“结巴分词”，对红楼梦前80回和后40回进行切词，并分别统计词汇出现的次数，其中剔除字符长度为1的词汇。然后分别打印前80回和后40回程序处理结果即打印词频。

得到实验数据便可对词频进行分析处理。利用表格呈现统计数据可以清楚地看出高频词使用习惯存在一定差异，重复率较低且相对数量相差较大（已对前后文本字数不一致进行补偿处理），初步判断前80回和后40回不是一人所著。然后利用matplotlib进行绘图，分别画出以前80回为对照的后40回词频重复量及数量（运行程序弹出的第一个图）及以后40回为对照的前80回词频重复量及数量（运行程序弹出的第二个图）根据对比可以看出，二者曲线相对弧度差距较大，较对照组均有明显断点，由此可以得出结论，前80回和后40回不是一人所著。利用机器学习及结果也可验证这一结论

1. 实验步骤（20分）

1.预处理，把回目编号删除，然后根据标点符号等，把每一句话标记出来；

2.利用开源分词组件“结巴分词”对红楼梦全书进行分词，统计词频，把出现100次以上的高频词语标记出来；

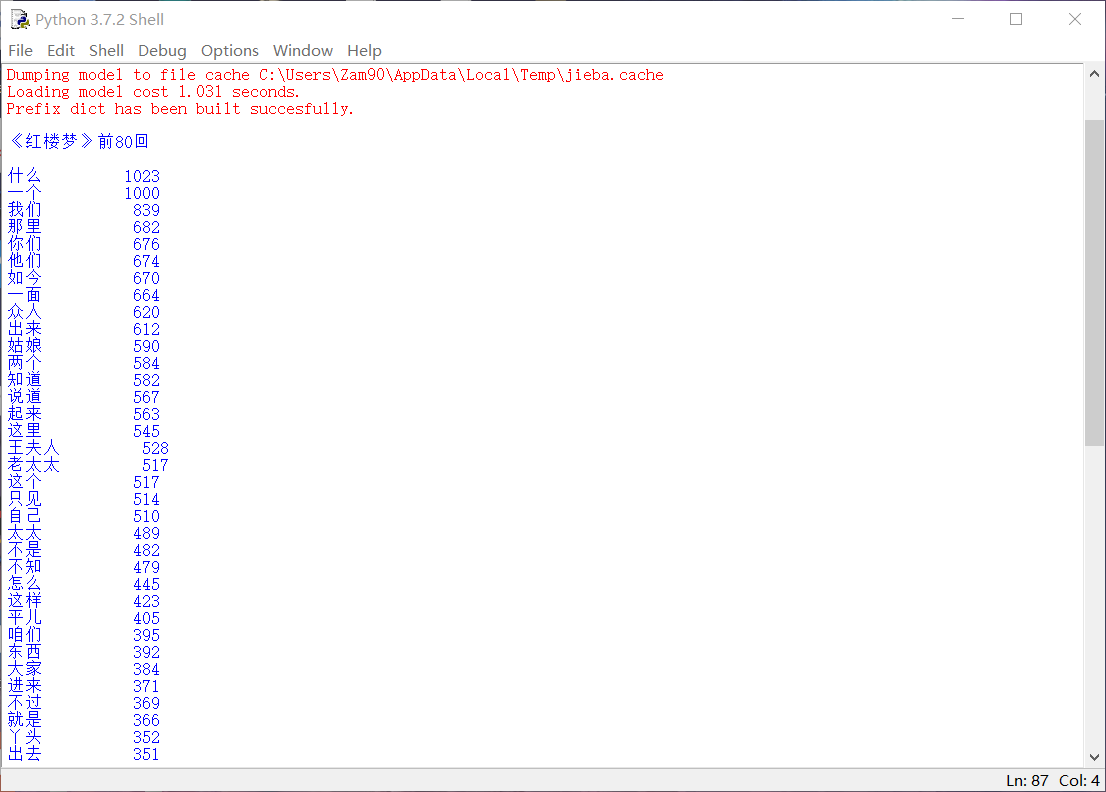
3.统计每一章高频词语出现的频次；

4.前80回、后40回各选15回分别作为训练数据，通过机器学习得出用词特征；

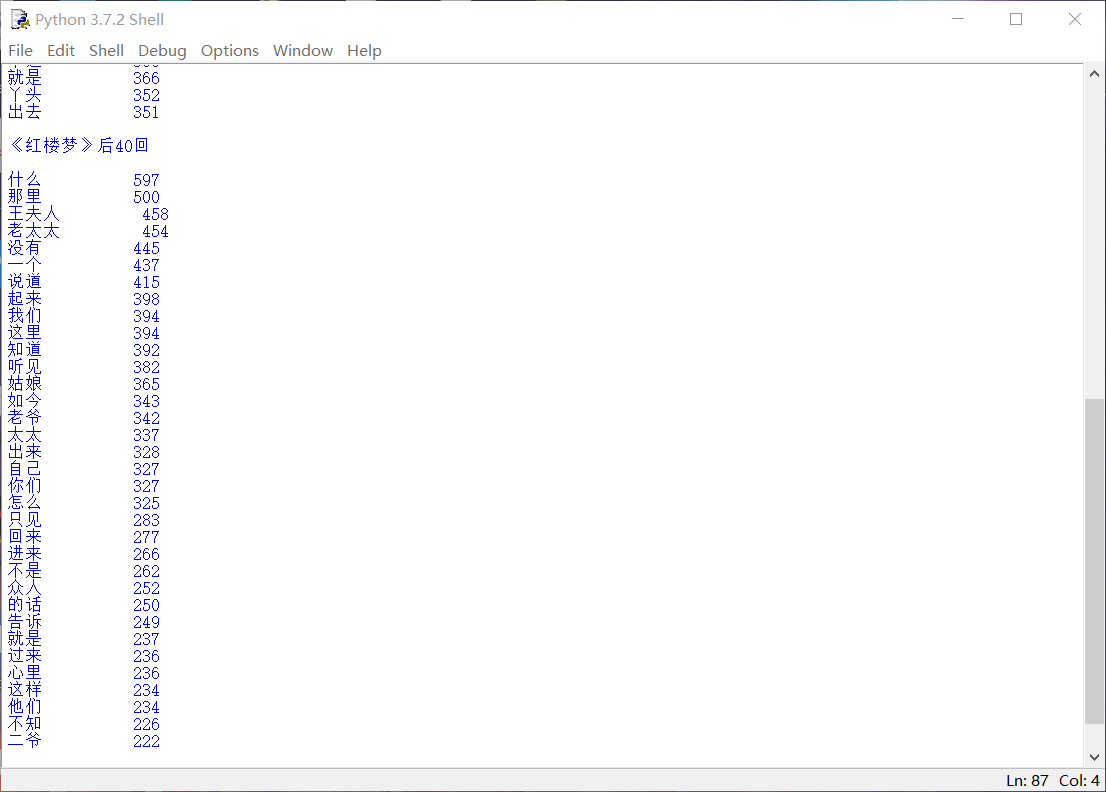
5.根据机器学习训练出的用词特征，计算每章是否和所属前80回或后40回相同，如果相同，则说明前80回和后40回很可能是两个人写的。换句话说，如果给出前80章的某一章，机器能算出它属于前80回而不是后40回；给出后40回的某一章，机器算出它属于后40回而不是前80回，则显然前80回和后0回遣词造句方面不同，从而得出不是同一个作者。

1. 程序结果截图（20分）

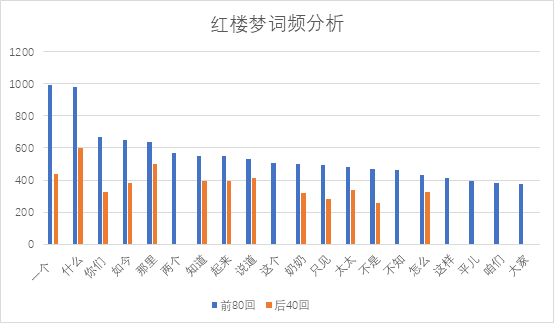
1.前80回各词语出现的频数（已去除人名）



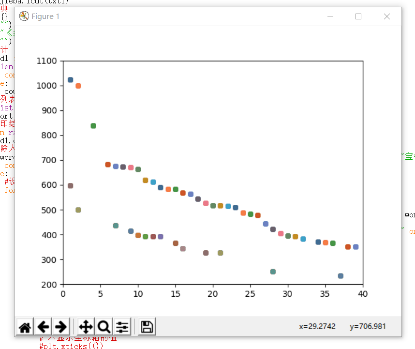
2.后40回各词语出现的频数（已去除人名）



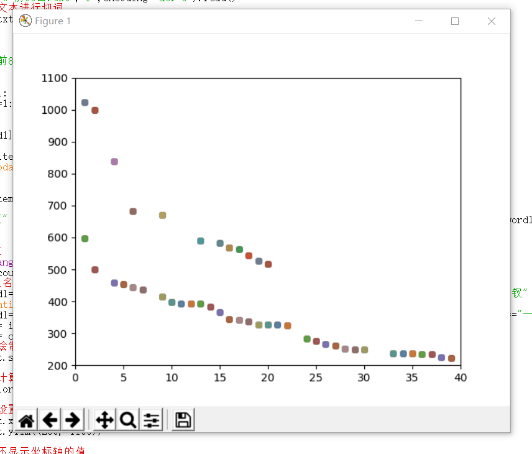
3.前80回与后40回高频词词频总量及数目对比分析



4.以前80回为对照 后40回词频重复量及数量（运行程序弹出的第一个图）



4.以后40回为对照 前80回词频重复量及数量（运行程序弹出的第二个图）



1. 程序源代码（20分）

#Python 3.7 64-bit

#author Zam90

#导入jieba库

import jieba

#导入sklearn库

from sklearn import metrics

import matplotlib.pyplot as plt

import numpy as np

import matplotlib.pyplot as plt

# 数据个数

n = 2000

#导入分析文本

txt1=open("《红楼梦》前80回.txt",'r',encoding="utf-8").read()

#利用jieba库对分析文本进行切词

words1=jieba.lcut(txt1)

#创建字典

count1={}

print ("")

print ("《红楼梦》前80回")

print ("")

#分析统计

for word1 in words1:

if len(word1)==1:

continue

else:

count1[word1]=count1.get(word1,0)+1

#转换成列表

items=list(count1.items())

items.sort(key=lambda x:x[-1],reverse=True)

#循环打印结果

for i in range(40):

word1,count1=items[i]

#剔除人名

if word1=="宝玉" or word1=="贾母" or word1=="凤姐" or word1=="贾琏" or word1=="黛玉" or word1=="宝钗" or word1=="袭人":

continue

else:

print("{0:<10}{1:>5}".format(word1,count1))

#设置x、y值

for i in range(40):

word1,count1=items[i]

#剔除人名

if word1=="宝玉" or word1=="贾母" or word1=="凤姐" or word1=="贾琏" or word1=="黛玉" or word1=="宝钗" or word1=="袭人":

continue

else:

x = i

y = count1

# 绘制散点图

plt.scatter(x, y, s = 30, alpha = 0.5)

# 计算颜色值

color = np.arctan2(y, x)

# 设置坐标轴范围

plt.xlim((0, 40))

plt.ylim((200, 1100))

# 不显示坐标轴的值

#plt.xticks(())

#plt.yticks(())

print ("")

print ("《红楼梦》后40回")

print ("")

#导入分析文本

txt2=open("《红楼梦》后40回.txt",'r',encoding="utf-8").read()

#利用jieba库对分析文本进行切词

words2=jieba.lcut(txt2)

#创建字典

count2={}

#分析统计

for word2 in words2:

if len(word2)==1:

continue

else:

count2[word2]=count2.get(word2,0)+1

#转换成列表

items=list(count2.items())

items.sort(key=lambda x:x[-1],reverse=True)

#循环打印结果

for i in range(40):

word2,count2=items[i]

#剔除人名

if word2=="宝玉" or word2=="贾母" or word2=="凤姐" or word2=="贾琏" or word2=="黛玉" or word2=="宝钗" or word2=="袭人":

continue

else:

print("{0:<10}{1:>5}".format(word2,count2))

#设置x、y值

for i in range(40):

word2,count2=items[i]

#剔除人名

if word2=="宝玉" or word2=="贾母" or word2=="凤姐" or word2=="贾琏" or word2=="黛玉" or word2=="宝钗" or word2=="袭人":

continue

if word2=="什么" or word2=="一个" or word2=="我们" or word2=="那里" or word2=="你们" or word2=="他们" or word2=="如今" or word2=="一面" or word2=="众人" or word2=="出来" or word2=="姑娘" or word2=="两个" or word2=="知道" or word2=="说道" or word2=="起来" or word2=="这里":

x = i

y = count2

# 绘制散点图

plt.scatter(x, y, s = 30, alpha = 0.5)

# 计算颜色值

#color = np.arctan2(y, x)

# 设置坐标轴范围

plt.xlim((0, 40))

plt.ylim((200, 1100))

# 不显示坐标轴的值

#plt.xticks(())

#plt.yticks(())

plt.show()

#导入分析文本

txt2=open("《红楼梦》后40回.txt",'r',encoding="utf-8").read()

#利用jieba库对分析文本进行切词

words2=jieba.lcut(txt2)

#创建字典

count2={}

#分析统计

for word2 in words2:

if len(word2)==1:

continue

else:

count2[word2]=count2.get(word2,0)+1

#转换成列表

items=list(count2.items())

items.sort(key=lambda x:x[-1],reverse=True)

#循环打印结果

for i in range(40):

word2,count2=items[i]

#剔除人名

if word2=="宝玉" or word2=="贾母" or word2=="凤姐" or word2=="贾琏" or word2=="黛玉" or word2=="宝钗" or word2=="袭人":

continue

else:

#设置x、y值

for i in range(40):

word2,count2=items[i]

#剔除人名

if word2=="宝玉" or word2=="贾母" or word2=="凤姐" or word2=="贾琏" or word2=="黛玉" or word2=="宝钗" or word2=="袭人":

continue

else:

x = i

y = count2

# 绘制散点图

plt.scatter(x, y, s = 30, alpha = 0.5)

# 计算颜色值

#color = np.arctan2(y, x)

# 设置坐标轴范围

plt.xlim((0, 40))

plt.ylim((200, 1100))

# 不显示坐标轴的值

#plt.xticks(())

#plt.yticks(())

#导入分析文本

txt1=open("《红楼梦》前80回.txt",'r',encoding="utf-8").read()

#利用jieba库对分析文本进行切词

words1=jieba.lcut(txt1)

#创建字典

count1={}

print ("")

print ("《红楼梦》前80回")

print ("")

#分析统计

for word1 in words1:

if len(word1)==1:

continue

else:

count1[word1]=count1.get(word1,0)+1

#转换成列表

items=list(count1.items())

items.sort(key=lambda x:x[-1],reverse=True)

#循环打印结果

for i in range(40):

word1,count1=items[i]

#剔除人名

if word1=="宝玉" or word1=="贾母" or word1=="凤姐" or word1=="贾琏" or word1=="黛玉" or word1=="宝钗" or word1=="袭人":

continue

else:

#设置x、y值

for i in range(40):

word1,count1=items[i]

#剔除人名

if word1=="宝玉" or word1=="贾母" or word1=="凤姐" or word1=="贾琏" or word1=="黛玉" or word1=="宝钗" or word1=="袭人":

continue

if word1=="什么" or word1=="那里" or word1=="王夫人" or word1=="老太太" or word1=="没有" or word1=="一个" or word1=="说道" or word1=="起来" or word1=="我们" or word1=="这里" or word1=="知道" or word1=="听见" or word1=="姑娘" or word1=="如今":

x = i

y = count1

# 绘制散点图

plt.scatter(x, y, s = 30, alpha = 0.5)

# 计算颜色值

color = np.arctan2(y, x)

# 设置坐标轴范围

plt.xlim((0, 40))

plt.ylim((200, 1100))

# 不显示坐标轴的值

#plt.xticks(())

#plt.yticks(())

plt.show()