

Homework1 报告

霍瑜 201605130119

第一步：分割单词，单词标准化，去停用词

问题 1：根据哪些字符来分割？

理想情况下在英文文档中，应该以空格，换行等空字符来分割，但在此数据集中存在标点符号后无空格的情况，这就会导致标点符号连接的两个单词会被分到一起，所以我们需要把标点符号也加入到分割符中。但这又会导致另一个问题，类似 U.S.A 的字符串，如果采用上述方法，会把 U.S.A 分割为三个单字符，也就损坏了信息。

但文档中标点符号后无空格的情况多与类似 U.S.A 的情况，所以综合考虑还是把标点符号也加入分隔符。

问题 2：如何单词标准化？

方案：去掉所有非拉丁字母的字符并把大写转小写。

本来我还想着保留数字，但之后发现保留数字的单词很杂乱，有很多无意义的内容

问题 3：去停用词

网上找了个停用词表，包含 891 个单词，直接相减

代码见 `get_word.py`

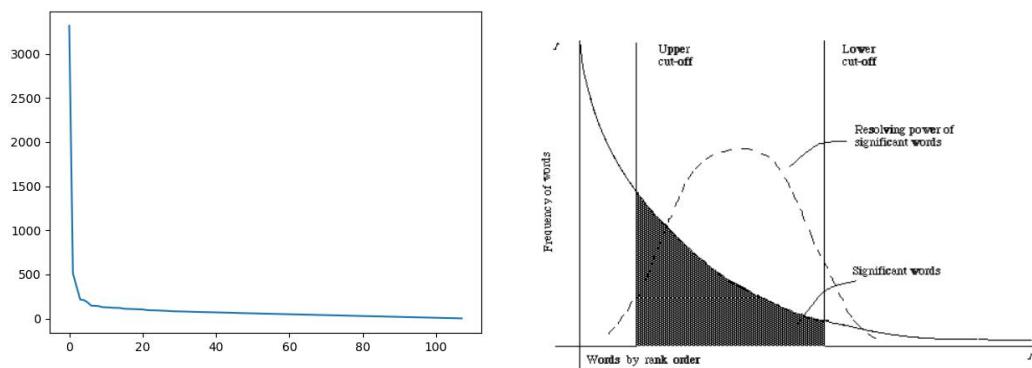
经过上述处理后的文档信息见 `article1.txt`，其中每行表示一个文档，每行由多个空格分开的单词组成

第二步：去除无用的单词

问题 1：Zipf's law

利用数据我验证了一下 Zipf's law。

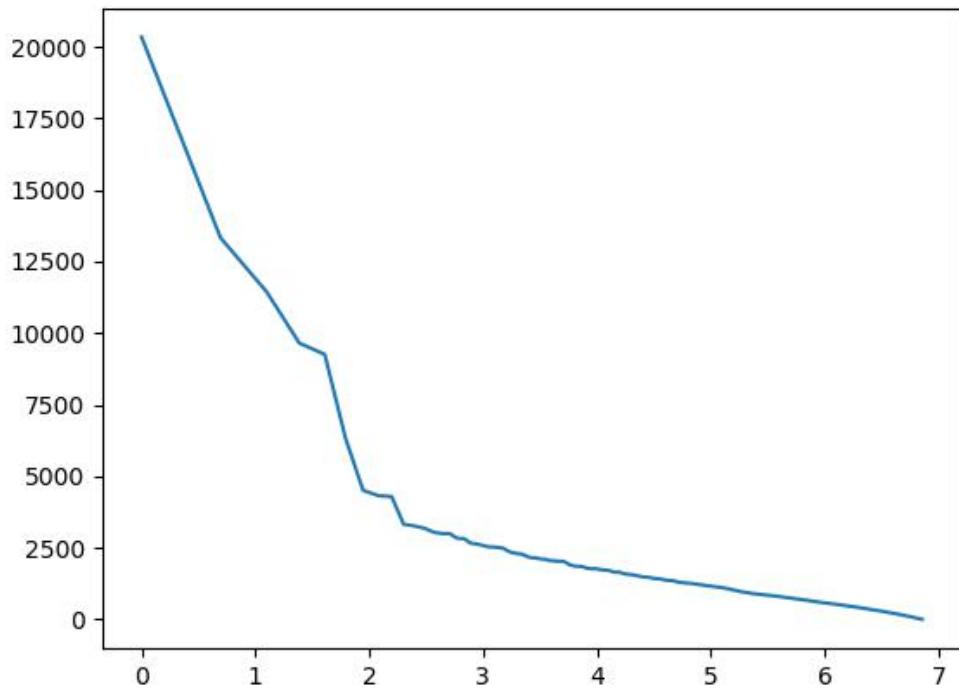
下图：横坐标表示单词词频的排名，纵坐标表示单词出现的次数
左：我跑出的图 右：课上 PPT 中的图



差别有点大！！！！

个人认为是数据量太小，导致很多的单词都只出现了少数次，才会显得曲线过早的逼近零

为扩大效果，我对横坐标取了 \log ，得到下图：



总的来看，还算 ok

问题 2：选取的词频范围？

高频单词：

[('subject', 20341), ('writes', 13332), ('article', 11444), ('dont', 9649), ('people', 9256), ('time', 6332), ('apr', 4510), ('god', 4321), ('system', 4288), ('maxaxaxaxaxaxaxaxaxaxaxax', 3321), ('i've', 3262), ('email', 3183), ('file', 3043), ('windows', 2999), ('question', 2966), ('read', 2834), ('government', 2818), ('bit', 2664), ('doesn't', 2632), ('david', 2586), ('program', 2535), ('drive', 2526), ('university', 2510), ('date', 2486), ('space', 2385), ('didn't', 2330), ('software', 2300), ('power', 2287), ('day', 2236), ('cs', 2172), ('game', 2153), ('true', 2148), ('law', 2117), ('john', 2105), ('set', 2090), ('uk', 2050), ('car', 2049), ('lot', 2040), ('computer', 2032), ('ac', 2031), ('org', 2015), ('list', 1967), ('support', 1897), ('real', 1887), ('key', 1861), ('jesus', 1860), ('life', 1846), ('hard', 1845), ('wrong', 1818), ('call', 1784), ('post', 1776), ('card', 1768), ('reason', 1767), ('called', 1765), ('files', 1764), ('gov', 1741), ('free', 1731), ('image', 1730), ('net', 1720), ('version', 1716), ('cc', 1696), ('team', 1692), ('isn't', 1653), ('news', 1652), ('person', 1648), ('public', 1645), ('send', 1644), ('mark', 1621), ('bad', 1597), ('heard', 1591), ('wrote', 1578), ('chip', 1578), ('bill', 1573), ('de', 1572), ('internet', 1559), ('sun', 1558), ('book', 1534), ('phone', 1529), ('nasa', 1518), ('times', 1515), ('systems', 1501), ('control', 1492), ('idea', 1481), ('christian', 1477), ('left', 1476), ('window', 1471), ('evidence', 1470), ('start', 1463), ('children', 1461), ('dos', 1444), ('ill', 1429), ('note', 1428), ('human', 1424), ('gun', 1423), ('word', 1416), ('remember', 1414), ('info', 1414), ('questions', 1405), ('michael', 1400), ('mail', 1380), ('change', 1378), ('mac', 1369), ('ftp', 1365), ('code', 1363), ('source', 1363), ('opinions', 1357), ('israel', 1356), ('games', 1354), ('graphics', 1354), ('based', 1350),

低频单词:

('charioteer', 1), ('shieldbearer', 1), ('asiatics', 1), ('wretches', 1), ('arrayed', 1), ('transgressed', 1), ('greatofstrength', 1), ('sayth', 1), ('spue', 1), ('fpuazbjuts', 1), ('agrjuts', 1), ('meadows', 1), ('inciting', 1), ('moromn', 1), ('eccommunicated', 1), ('nonmormons', 1), ('leftlefthanded', 1), ('legitimized', 1), ('accountable', 1), ('dontidt', 1), ('sinss', 1), ('hylomorphism', 1), ('kindergardeners', 1), ('pedestal', 1), ('explainable', 1), ('optomologists', 1), ('grayshaded', 1), ('exerted', 1), ('indentifiable', 1), ('niles', 1), ('housecleaning', 1), ('biologys', 1), ('isthese', 1), ('textpxl', 1), ('permeating', 1), ('illiteracy', 1), ('eldridges', 1), ('wordstoknow', 1), ('recuperate', 1), ('wirebound', 1), ('dbaandrew', 1), ('sunstone', 1), ('plainness', 1), ('biblecentricty', 1), ('sixseven', 1), ('biblederived', 1), ('interpeting', 1), ('people', 1), ('wahr', 1), ('thout', 1), ('preeaster', 1), ('posteaster', 1), ('referencs', 1), ('buggenerator', 1), ('cprogrammers', 1), ('editdruntrit', 1), ('falsh', 1), ('surviors', 1), ('unverified', 1), ('bders', 1), ('parnoia', 1), ('cornuers', 1), ('gskinneruiuc', 1), ('sharpennesis', 1), ('aberrations', 1), ('equiped', 1), ('astrologers', 1), ('sexists', 1), ('neofruedian', 1), ('reponsible', 1), ('naturalists', 1), ('novakadvtech', 1), ('neesecerritos', 1), ('differences', 1), ('everburning', 1), ('unseat', 1), ('mutinous', 1), ('hinniom', 1), ('abhorrence', 1), ('josiah', 1), ('dispised', 1), ('aceldama', 1), ('miracleworking', 1), ('kindle', 1), ('incorerigibly', 1), ('writh', 1), ('mindedness', 1), ('thot', 1), ('bakerjtgtpbx', 1), ('cunv', 1), ('jgsncratl', 1), ('rifleaseassociation', 1), ('yearns', 1), ('tradingselling', 1), ('rcfinnrlemx', 1), ('digi', 1), ('palodigi', 1), ('massadas', 1), ('inquisitorial', 1), ('bfhdarkside', 1), ('beomes', 1)]

finished in 2.5s

由于已经去除了停用词，也就可以假定不存在词频特高且无用的单词，于是我就只对下限做了限制，综合词典大小，词意等方面考虑，最后选择去除词频小于等于 10 的单词

词典大小降了 8.5 倍左右

170303
20616
[Finis]

代码见 get_word1.py

经过上述处理后的文档信息见 article2.txt

第三步：词形还原，词干提取

使用 NLTK 处理库

问题 1: 选择词形还原还是词干提取？

简单试了试 NLTK 库里的 stemming 和 lemmatization，发现词形还原的效果很差，连简单的 drove, driving 都处理不了，于是果断选区词干提取。

代码见 `get_word2.py`

经过上述处理后的文档信息见 `article3.txt`

第四步：TF-IDF 计算

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

$$idf_i = \lg \frac{|D|}{|\{j : t_i \in d_j\}|}$$

利用公式计算出 `tf`, `idf`, 直接相乘

代码见 `get_vector.py`

经过上述处理后的文档信息见 `vector.txt` (由于太大, 未上传到 `github`)

问题: 词典两万多个单词, 那么表示一个文档的向量就有两万维, 导致很多都是零, 这是 VSM 很大的一个问题

第五步：计算文档相似度（即向量距离）

$$\text{cosine}(d_i, d_j) = \frac{V_{d_i}^T V_{d_j}}{\left|V_{d_i}\right|_2 \times \left|V_{d_j}\right|_2}$$

采用向量夹角 `cos` 值表示, 尝试计算了一下第一个文档和第二个文档的相似度:

```
22 print(calc(vector[0], vector[1], len(vector[0])))
23
0.39727246021557594
[Finished in 15.1s]
```

代码见 `calc_distance.py`