实验二 实验报告

姓名：单宝迪 学号：201700210069 班级：17数据

实验环境和实验时间

实验环境：

- 硬件环境: Intel(R) Core(TM) i7-8550U 16GRAM
- 软件环境: Windows 10 专业版 Python3.7
- IDE: Pycharm Jupyter-Notebook

实验时间：

- 项目创建时间 2019.9.27
- 项目结束时间 2019.10.9
- 项目报告提交时间 2019.10.9

实验目标

- 在Homework1.1的基础上实现最基本的Ranked retrieval model
- Use SMART notation: lnc.ltn
- 改进Inverted index

实现过程

1.建立倒排索引

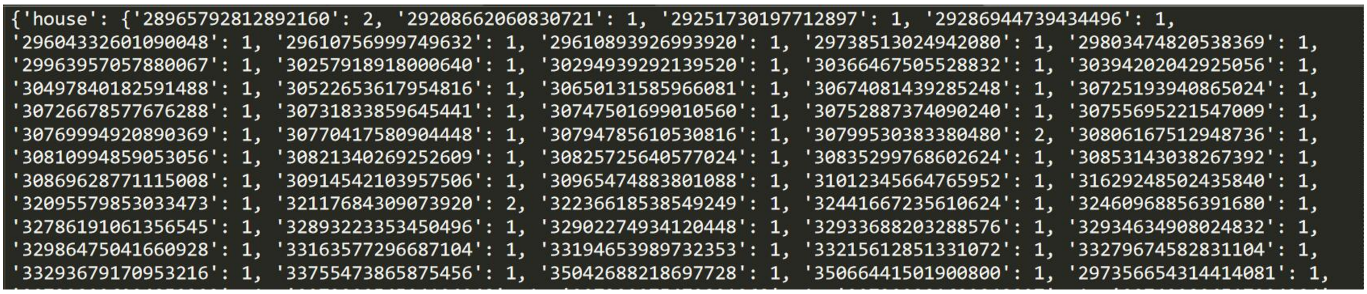
相比于Homework1.1,本次作业的倒排索引需要将doc，变为<docid, td>.因此，倒排索引在之前的基础上做了改进，实现源码如下:

```
x = open('file/word.txt', 'w')
for line in f:
    word = TextBlob(line).words.singularize()
    word[0] = Word(word[0])
    # word[0]是 tweet id
    for i in word[1:]:
        # i=Word(i)
        if i not in Dict:
            #tmp={word[0]:1}
            Dict[i]={}
            Dict[i][word[0]] = 1
        else:
            if word[0] not in Dict[i]:
                Dict[i][word[0]] = 1
            else:
                Dict[i][word[0]]=Dict[i][word[0]]+1

# print(Dict['may'])
```

```
x.write(str(Dict))
x.close()
```

倒排索引结果示例如下:



```
{'house': {'28965792812892160': 2, '29208662060830721': 1, '29251730197712897': 1, '29286944739434496': 1,
'29604332601090048': 1, '29610756999749632': 1, '29610893926993920': 1, '29738513024942080': 1, '29803474820538369': 1,
'29963957057880067': 1, '30257918918000640': 1, '30294939292139520': 1, '30366467505528832': 1, '30394202042925056': 1,
'30497840182591488': 1, '30522653617954816': 1, '30650131585966081': 1, '30674081439285248': 1, '30725193940865024': 1,
'30726678577676288': 1, '30731833859645441': 1, '30747501699010560': 1, '30752887374090240': 1, '30755695221547009': 1,
'30769994920890369': 1, '30770417580904448': 1, '30794785610530816': 1, '30799530383380480': 2, '30806167512948736': 1,
'30810994859053056': 1, '30821340269252609': 1, '30825725640577024': 1, '30835299768602624': 1, '30853143038267392': 1,
'30869628771115008': 1, '30914542103957506': 1, '30965474883801088': 1, '31012345664765952': 1, '31629248502435840': 1,
'32095579853033473': 1, '32117684309073920': 2, '32236618538549249': 1, '32441667235610624': 1, '32460968856391680': 1,
'32786191061356545': 1, '32893223353450496': 1, '32902274934120448': 1, '32933688203288576': 1, '32934634908024832': 1,
'32986475041660928': 1, '33163577296687104': 1, '33194653989732353': 1, '33215612851331072': 1, '33279674582831104': 1,
'33293679170953216': 1, '33755473865875456': 1, '35042688218697728': 1, '35066441501900800': 1, '297356654314414081': 1,
```

2.计算每篇doc的cosine值

cosine的计算公式为:

$c(cosine)=\frac{1}{\sqrt{w_1^2+w_2^2+\cdots+w_M^2}}$

考虑到计算每个doc的cosine的计算量较大，如果再query时计算，对查询速度有影响，因此，我采用了一次计算出所有文本的cosine值导入文件的方法，process代码如下：

```
S=open('file/cosinelog.txt','w')
Dict1 = {}
Cos={}
for line in f:
    word = TextBlob(line).words.singularize()
    word[0] = Word(word[0])
    # word[0]是 tweet id
    Dict1[word[0]] = {}
    for i in word[1:]:
        # i=Word(i)
        if i not in Dict1[word[0]]:
            Dict1[word[0]][i] = 1
        else:
            Dict1[word[0]][i] = Dict1[word[0]][i] + 1

for i in Dict1:
    ans = 0
    for word in Dict1[i]:
        tmp=1+math.log10(int(Dict1[i][word]))
        ans += tmp**2
    ans=math.sqrt(ans)
    print(ans)
    Cos[i]=ans

S.write(str(Cos))
```

3.计算结果

3.1 计算wtq

考虑到查询方式为lnc,ltn,故需要对query中的词频求log，并乘以其idf。

计算公式为：

$$l(logarithm)=1+log(tf_{t,d})$$

$$t(idf)=log\frac{N}{df_t}$$

具体函数实现如下：

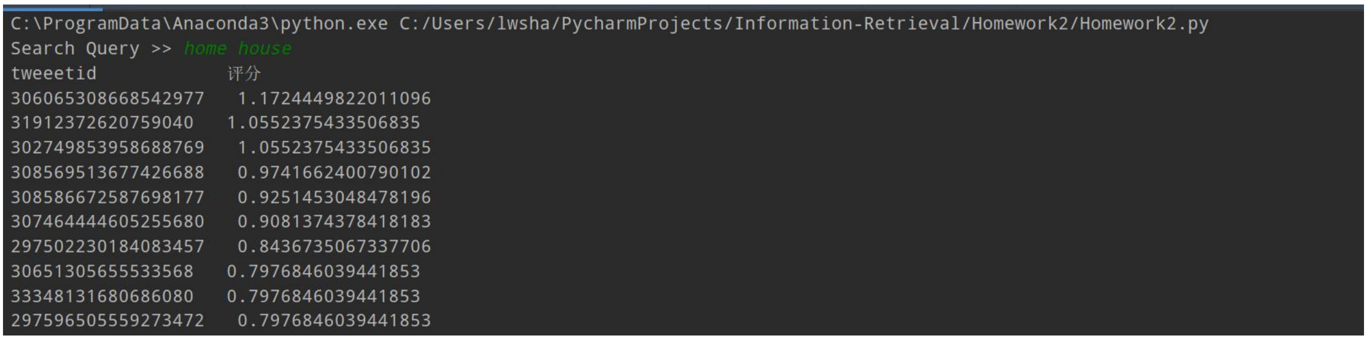
```python
def wtq(terms, term):
    global Dict
    num = 0
    for i in terms:
        if i == term:
            num += 1
    idf = math.log10(N / len(Dict[term]))
    wtq =  1 + math.log10(num)
    return idf * wtq
```

3.2 查询函数

对于doc中wtd的计算，由于计算量较小，我们将求log和除以length的过程整合到了search函数中。 Search函数的实现如下：

```python
def Search(terms):
    getDict()
    score = {}
    for w in terms:
        Wtq = wtq(terms, w)
        for i in Dict[w]:
            td = int(Dict[w][i])
            wtd = 1 + math.log10(td)
            if i not in score:
                score[i]=wtd*Wtq
            else:
                score[i]+=wtd*Wtq
    for doc in score:
        score[doc]=score[doc]/cos[doc]
    result = sorted(score.items(), key=lambda x: x[1], reverse=True)
    print("tweeetid            评分")
    for i in result[:10]:
        print(str(i[0])+"   "+str(i[1]))
```

运行示例



```
C:\ProgramData\Anaconda3\python.exe C:/Users/lwsha/PycharmProjects/Information-Retrieval/Homework2/Homework2.py
Search Query >> home house
tweeetid             评分
306065308668542977    1.1724449822011096
31912372620759040    1.0552375433506835
302749853958688769    1.0552375433506835
308569513677426688    0.9741662400790102
308586672587698177    0.9251453048478196
307464444605255680    0.9081374378418183
297502230184083457    0.8436735067337706
30651305655533568    0.7976846039441853
33348131680686080    0.7976846039441853
297596505559273472    0.7976846039441853
```

反思与感悟

通过本次实验，对于倒排索引的构建有了更充分的认识，对于*SMART notation*有了更深的了解。

备注：Jupyter Notebook文件只是中间形式，实验结果以py文件为准。