实验一 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 2019.9.20
- 项目结束时间 2019.9.24
- 项目报告提交时间 2019.9.27

实验目标

- 在tweets数据集上构建Inverted index
- 实现布尔查询。Boolean Retrieval Model：And, Or ,Not
- 进行查询优化：拓展查询词汇数量

实现过程

Step1 倒排索引的建立

首先，将源数据中的text与tweet id提取出来，为了后续的运行速率，将提取出的数据写入文件中，便于后续读取。Step1的代码如下：

```
f = open('tweets.txt', 'r')
x = open('text.txt', 'w')
  for i in f:

    #得到text

    pr1 = i.split(', "text": "')
    line = pr1[1].split('", "timeStr"')
    text1 = line[0]+"\n"

    #得到id

    pr2 = i.split(', "tweetId": "')
    line = pr2[1].split('", "errorCode": "')
    id = line[0]
    x.write(id+" "+text1.lower())
```

```
f.close()
x.close()
```

然后，我们以word作为key，docid列表作为value，以字典的形式生成和储存倒排索引。同时，通过TextBlob库，对倒排索引的结果进行处理，得到最终版的倒排索引。

```
Dict = defaultdict(dict)


def makeDict():
    global Dict

    f = open('file/text.txt', 'r')
    x = open('file/word.txt', 'w')

    for line in f:
        word = TextBlob(line).words.singularize()
        word[0] = Word(word[0])
        for i in word[1:]:

            if i not in Dict:
                Dict[i] = []
                Dict[i].append(word[0])
            else:
                Dict[i].append(word[0])
    for i in Dict:
        Dict[i].sort()

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

Step2 编写布尔查询语句

编写布尔查询的语句，实现两个词的And，Or，Not查询

```
def And(term1, term2):
    global Dict
    answer = []
    if (term1 not in Dict) or (term2 not in Dict):
        return answer
    else:
        i = len(Dict[term1])
        j = len(Dict[term2])
        x = 0
        y = 0
        l1 = Dict[term1]
        l2 = Dict[term2]
        while x < i and y < j:
            if l1[x] == l2[y]:
                answer.append(l1[x])
                x += 1
                y += 1
            elif l1[x] < l2[y]:
                x += 1
            else:
                y += 1
        return answer


def Or(term1, term2):
    global Dict
    answer = []
    if (term1 not in Dict) or (term2 not in Dict):
        return answer
    else:
        answer = Dict[term1] + Dict[term2]
        return answer


def Not(term1, term2):
    global Dict
    answer = []
    if term1 not in Dict:
        return answer
    elif term2 not in Dict:
        answer = Dict[term1]
        return answer

    else:
        answer = Dict[term1]
        ANS = []
        for ter in answer:
            if ter not in Dict[term2]:
                ANS.append(ter)
        return ANS
```

3.查询优化

拓展程序，使程序可查询的单词数量达到三个。**特别注意，三个词查询时，需要考虑and和or的顺序。**

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