

BOĞAZİÇİ UNIVERSITY

CMPE 493

ASSIGNMENT 1

SPRING 2018

A Simple Document Retrieval System for Boolean Queries

Mustafa Enes ÇAKIR

March 8, 2018

1 Data Preprocessing

Indexer class handles all index operations. It splits corpus by */REUTERS/* for dividing it to documents. Then it extracts *news ID*, *title* and *body* for each document. It concatenated title and body of new for tokenization. *Tokenizer* class handles all data preprocessings. First of all, it makes all text lowercase. Secondly, it replaces punctuation with space. So user can find *six-week-old* word using *six*, *week* or *old*. After, it removes digits and extra whitespaces such as double adjacent spaces. It splits processed text by whitespaces and stems each token. Finally it put them to inverted index based on news ID and positions.

(a) How many tokens does the corpus contain before stopword removal and stemming? 2737964

(b) How many tokens does the corpus contain after stopword removal and stemming? 2126652

(c) How many terms(unique tokens) are there before stopword removal, stemming, and case - folding? 129173

(d) How many terms(unique tokens) are there after stopword removal, stemming, and casefolding? 99591

(e) List the top 20 most frequent terms before stopword removal, stemming, and casefolding?

('the', 119584)
('of', 72137)
('to', 68413)
('and', 53275)
('in', 49814)
('a', 48193)
('said', 35721)
('for', 25120)
('mln', 24805)
('The', 22828)
('', 19043)
('on', 17696)
('it', 17497)
('is', 16645)
('said.', 15784)
('dlrs', 15434)
('from', 14919)
('that', 14837)
('its', 14715)
('vs', 14599)

(f) List the top 20 most frequent terms after stopwords removal, stemming, and case - folding?

```
('to', 72528)
('said', 35919)
('mln', 25939)
('reuter', 19553)
('dlr', 19431)
('&#3;', 19043)
('said.', 15787)
('from', 15242)
('pct', 14843)
('vs', 14827)
('at', 14399)
('bank', 10220)
('ha', 10165)
('billion', 10083)
('u.s.', 9325)
('ct', 9259)
('would', 9156)
('compani', 8531)
('not', 8206)
('year', 8128)
```

2 Data Structures

My dictionary is a basic *Python dictionary*. It's key is stem and value is postings list. Each posting list is also *dictionary*. It's keys are documents ID and values are positions list as *Python list*.

```
inverted_index = {
    "stem1" : {docID1:[1,2,4], docID2: [3,5] .....},
    "stem2" : {docID2: [4,8] ...},
    .....
}
```

Code 1: Basic Data Structure

I split it to two parts, because we have to create two files. First file is *dictionary*. Keys are stems and values are postings list IDs.

```
dictionary = {  
    "stem1" : postingListID1,  
    "stem2" : postingListID2,  
    "stem3" : postingListID3  
}
```

Code 2: First File: Dictionary

Second file is also a *dictionary*. Keys are postings list IDs and values are postings list dictionary.

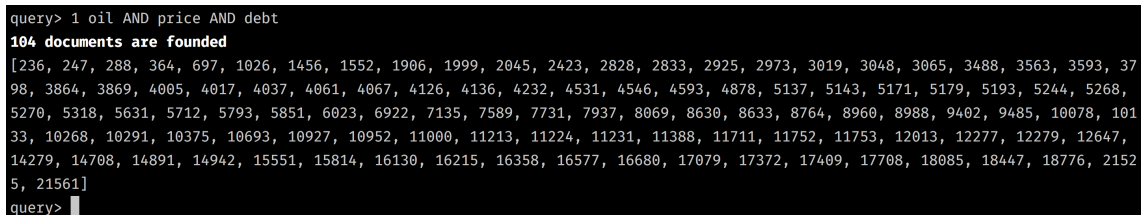
```
inverted index = {  
    postingListID1 : {docID1:[1,2,4], docID2: [3,5] .....},  
    postingListID1 : {docID2: [4,8] ...},  
    .....  
}
```

Code 3: Second File: Inverted Index

If user wants postings list of *car*, first it looks dictionary for postings list ID. After getting ID, it gets postings list based on the ID.

3 Screenshots

3.1 Conjunctive Query



```
query> 1 oil AND price AND debt  
104 documents are founded  
[236, 247, 288, 364, 697, 1026, 1456, 1552, 1906, 1999, 2045, 2423, 2828, 2833, 2925, 2973, 3019, 3048, 3065, 3488, 3563, 3593, 37  
98, 3864, 3869, 4005, 4017, 4037, 4061, 4067, 4126, 4136, 4232, 4531, 4546, 4593, 4878, 5137, 5143, 5171, 5179, 5193, 5244, 5268,  
5270, 5318, 5631, 5712, 5793, 5851, 6023, 6922, 7135, 7589, 7731, 7937, 8069, 8630, 8633, 8764, 8960, 8988, 9402, 9485, 10078, 101  
33, 10268, 10291, 10375, 10693, 10927, 10952, 11000, 11213, 11224, 11231, 11388, 11711, 11752, 11753, 12013, 12277, 12279, 12647,  
14279, 14708, 14891, 14942, 15551, 15814, 16130, 16215, 16358, 16577, 16680, 17079, 17372, 17409, 17708, 18085, 18447, 18776, 2152  
5, 21561]  
query> █
```

Figure 1: Conjunctive Query

3.2 Phrase Query

```
query> energy price
27 documents are founded
[1152, 1230, 2074, 4744, 4944, 5037, 5061, 5793, 5830, 5868, 6121, 8098, 10385, 10553, 10567, 10670, 11243, 11444, 11711, 12521, 12939, 16783, 17128, 17161, 17220, 17409, 19478]
query> █
```

Figure 2: Phrase Query

3.3 Proximity Query

```
Type help for any documentation
query> apple /5 computer
18 documents are founded
[361, 367, 385, 487, 669, 823, 1116, 2831, 3177, 4588, 5302, 11050, 12871, 13002, 17851, 19195, 19828, 21407]
query> government /3 government
30 documents are founded
[835, 1275, 2775, 4037, 4126, 4158, 5252, 5554, 5817, 5973, 6099, 7560, 8092, 8132, 8752, 10725, 11768, 11869, 12461, 14813, 15488, 16144, 16935, 16957, 17305, 17364, 18250, 18961, 19164, 19266]
query> debt /10 price /8 oil
2 documents are founded
[4878, 7589]
query> █
```

Figure 3: Proximity Query