1. How to use

My program put the boolean query and vector space cosine similarity together. The inverted index will be generated only once while you do the query.

When you run the program, after finishing the inverted index, which might take 1-3 minutes if you let it index all the files in the dataset.

You will see this on the screen:

```
Reading txt file   // Indicate the program is reading all the files into memory
Building inverted index. This may take about 1-2 minute // Doing the indexing
finish // Finish index

choose option:
1. Boolean Query[Press 1]
2. Vector Space Query[Press 2]
Input exit to terminate[Type 'exit']
Make your choice
```

If you input 1, and 2 the program will do boolean and vector space query respectively. If you input “exit”, the program will exit.

2. The structure of my inverted index

My inverted index contains the term and the frequency of the term in each documents. All the inverted index dictionary is stored in variable : *inverted*
For example:
inverted[‘information’] will show:

{'a9900431.txt': 1, 'a9900767.txt': 2, 'a9900117.txt': 2……}

This indicate term ‘information’ appears in documents ‘a9900431.txt’, ‘a9900767.txt' and so on also indicates how many times it appears in each document. So ‘a9900431.txt’: 1 means it appear in document ‘a9900431.txt’ one time.

3. Vetor Space Query
def get_raw_TF(inverted,queries) is used to get the raw_tf of the queries.
def getTF(inverted) is used to get the TF of each documents
def getIDF(N,inverted) is used to get the IDF of each term

def getTF_IDF(N,inverted) is used to get the TF_IDF in the vector space. It will use the result of getTF and getIDF

def getCosineSim(N,inverted,queries) This is the function to get the Cosine Similarity in vector space.

4 Reference
For how to tokenize the text, I use this way in Github as reference https://github.com/matteobertozzi/blog-code