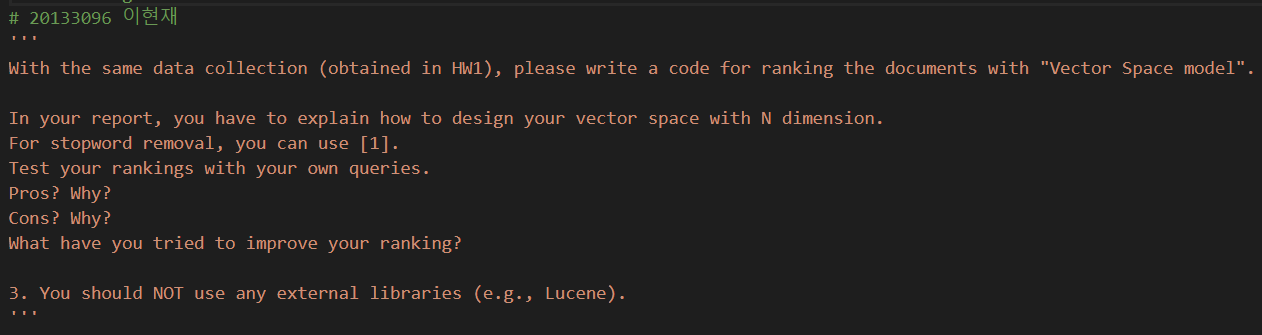


**자연어처리 및 정보검색 02분반 – Homework 2**

Student ID: 20133096

Student Name: 이현재

**프로젝트 목표:**

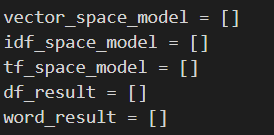


**프로젝트 설명:**

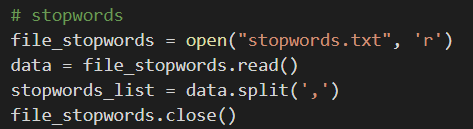
1. 입력 텍스트파일 (Query)와 주어진 텍스트 파일(Document 1 ~ 10) 까지의 유사도(similarity)를 측정하여 ranking 한다.
2. TF (Term frequency)는 특정한 단어가 문서 내에 얼마나 자주 등장하는지를 나타내고, DF(Document frequency)는 단어가 문서내에서 등장했는지를 나타낸다. 따라서 TF-IDF 가중치를 사용하기위해 DF의 역수인 IDF(Inverse Document Frequency)를 사용한다.
3. **외부라이브러리 사용의 제한으로 Python의 built-in 라이브러리인 파일입출력을 위한 os와 log 및 계산을 위한 math 함수만을 사용하였다.**

**프로젝트 과정:**

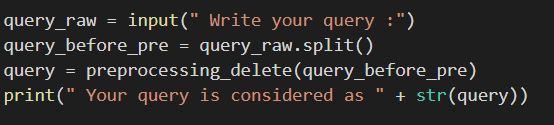
1. Vector space model을 정제된 query단어의 수 X 문서의 수 크기의 매트릭스로 정의한다. 그리고 이를 구하기 위해, idf space model과 tf space model를 정의한다. 또한, cos similarity 계산을 위해 df의 값들을 담을 df\_result와 단어가 제대로 전처리 되었는지 확인하기 위해 word\_result를 정의하였다.



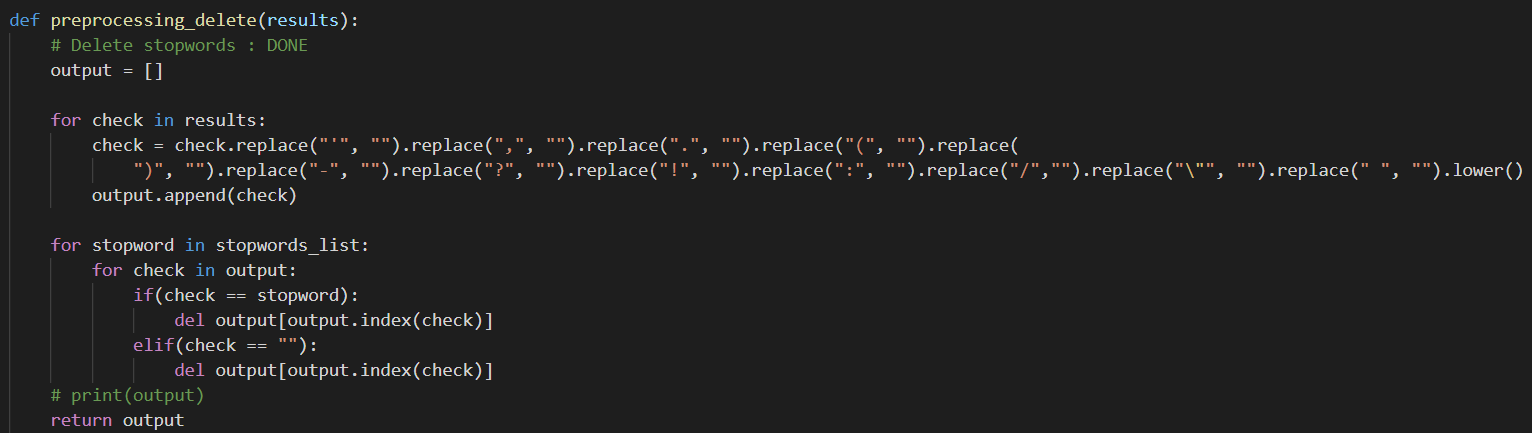
1. Stopwords를 구분하여 저장한다.



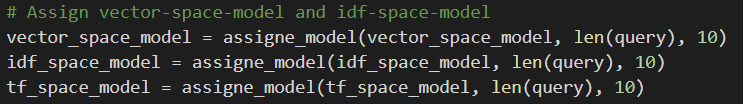
1. 사용자로부터 Query를 입력받는다.

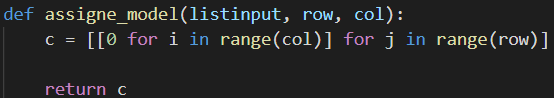


1. 입력받은 Query를 preprocessing\_delete 함수를 통해 전처리 과정을 거친다. Query내에 세미콜론,느낌표 등과 같은 문자들을 제거하고, 소문자로 변환 후 stopwords를 제거한다.

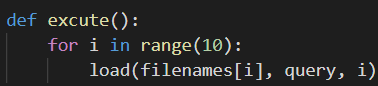


1. (1)에서 정의한 3개의 모델을 초기화해준다.





1. Query를 전처리 했으니 다음으로는 10개의 문서를 전처리 하여 그 결과를 idf space model, tf\_space\_model, word\_result, df\_result에 입력한다.



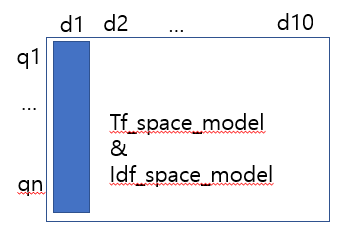
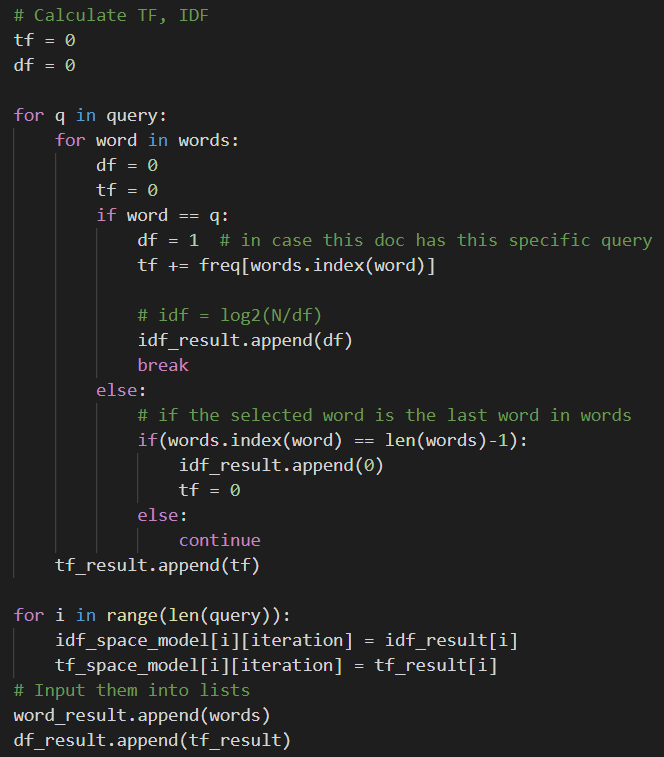
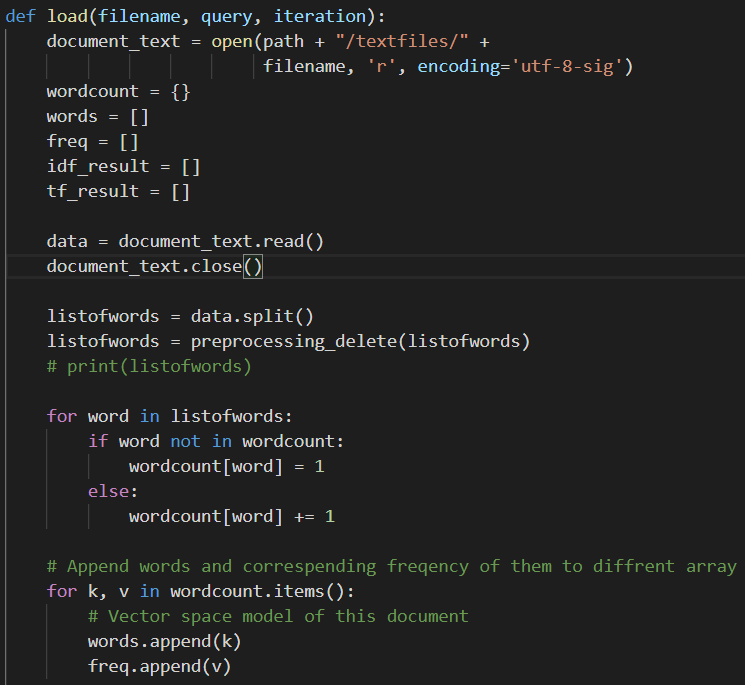
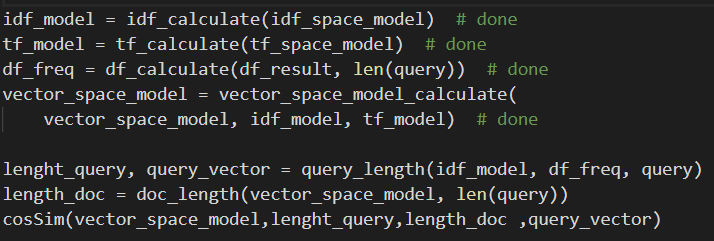


Figure Matrix 1

1. 먼저 첫번째 텍스트파일의 전처리과정을 거친다(preprocessing\_delete)
2. Listofwords에 해당 단어들을 담고, 각 단어의 횟수를 세서 wordcount에 담는다.
3. Wordcount를 각각 단어만 담은 word와 그 단어의 횟수를 담은 freq으로 나눈다.
4. Query의 첫번째 query word부터하여 그 query word가 문서의 단어들을 담은 word 리스트에 있는지 검사하고, 있으면 그 단어의 횟수에 대응하는 freq 리스트 요소 값을 tf에 담는다. 동시에 해당 문서에 해당 query 단어가 있음을 나타내기 위한 df를 1로 설정하고 idf\_result를 담는다.
5. 그리고 Matrix 1 그림은 idf\_space\_model과 tf\_space\_model에 해당 값들을 문서별로 담기 위해, 각 Iteration에 어떻게 작동하는지 나타낸다.
6. Query 및 10개의 텍스트문서에 전처리 과정이 끝나고, idf\_space\_model과 tf\_space\_model에 모든 값들이 입력이 됐으면, 각 값들을 계산해준다. 아래의 코드는 execute()이후 실행되는 모든 코드이다.



1. 먼저, idf\_calculate이다.

