INFORMATION RETRIEVAL (CS F469) ASSIGNMENT 1

Domain Specific Information Retrieval System

Members

Ahraz Rizvi 2015A7PS0012H

Mudit Agarwal 2015A7PS0117H

Rajkaran Sharma 2015A7PS0132H

Aditya Kashilkar 2015A7PS0098H

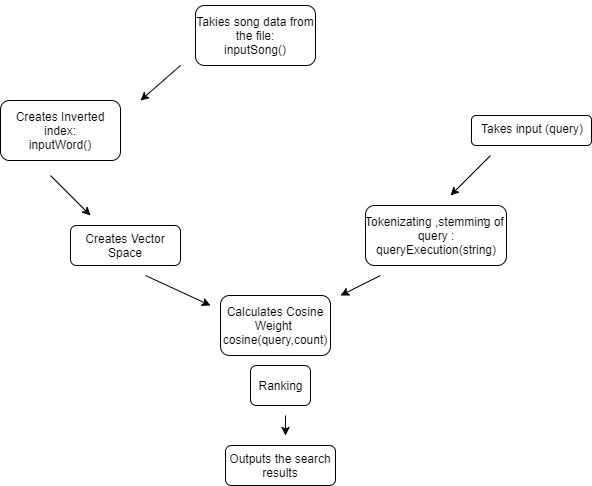
GETTING DATASET:

The dataset is taken from Kaggle. Dataset contains more than 380000 song lyrics in .csv format.

MAJOR DATA STRUCTURES USED

* ArrayList of object(word , hashmap ){ **ArrayList<word Object>** } – This is to form inverted index. Our object consists of a word and a hashmap where word is a string and hashmap contains list of objects consisting of a key and its value. Here key denotes the document id and value contains count of no. of words the document corresponding to key value contains.
* Array of Objects – To store the various details of songs like artist, release year, lyrics and other details.
* ArrayList of strings{ **ArrayList<string>**} - To store stopwords.
* Arrays – To store query tokens and counts. And also to store cosine weights of documents corresponding to query.

DESIGN ARCHITECTURE:



BRIEF NOTES ON CODE EXECUTION:

* First we take input from the dataset and stores documents as objects and an inverted index is created.
  + inputSong( ): This functions takes input from the dataset and stores it in an arraylist of objects. Here each song(document) is stored as an object.
  + inputWord( ): This function will form the inverted index from the arraylists of songs(documents) after tokenization and stemming and stop words removel.
* Then we calculate tf-idf from the inverted index and form vector space.
* After that we take input from the query and then we process query to make vectors and calculate cosine weights for each document.
  + queryExecution(String qer): Processes the query and calculates weights for cosine values for all documents. This also returns document with the highest cosine value.
    - Cosine(int q[ ][ ],int size): Calculates cosine value from if-idf vectors for query and documents.

RUNNING TIME

* For input dataset: 3.022 s
* For Preprocessing: 8.47 s
* For Search Retrieval: 0.2 sec