ASSIGNMENT-2

SWAGATAM CHAKRABORTI(MT18146)

1. **Tf-idf based document retrieval**

PREPROCESSING:

1. Each line of the file is pre processed individually
2. Word tokenization is done using nltk library using the regextokenization which handles the formation of the tokens and also the removal of the punctuations
3. Stopwords have been removed from the tokens formed.
4. Lemmatization have been performed over the tokens using the nltk library
5. If the line contains any numbers, it is converted into words using inject library and is stored in the vocab along with the number itself

METHODOLOGY:

1. Traverse through all the documents, preprocess the data and maintain a vocab dictionary for each documents having word as the key and corresponding term frequency of the word in the document.
2. Create a list of dictionaries, having document name as the key for each dictionary and vocab dictionary of the corresponding calculated in the previous step as the value.
3. For the query entered by the user is pre-processed and a processed query is obtained.
4. The query is then matched against the titles of the documents present in the index.html document. If there is 30% match of the query words with the title of the documents, the documents name is appended in the final document list.
5. For each words in the query, calculate the inverse document frequency, fot each documents, if the word matches in the vocab of the document, the tf-idf score is multiplied and appended in the dictionary with the document name as the key and tf-idf score as the value
6. Finally the dictionary is sorted based on the if-idf score.
7. For displaying the top k relevant documents, firstly the documents present in the match with the title displayed and remaining documents among the k documents is displayed from the sorted dictionary of document tf-idf score.

OBSERVATION

|  |  |  |  |
| --- | --- | --- | --- |
| SN.NO | QEURY | NUMBER OF RETRIEVAL | RETRIEVED DOCUMENTS |
| 1 | Disco can be fun | 5 | disco.be.fun,  discocanbefun.txt, cybersla.txt, vgilante.txt, gulliver.txt |
| 2 | Going 100 west by 53 north | 5 | 100west.txt, dakota.txt, vgilante.txt, gulliver.txt, batlslau.txt |

1. **Tf-idf using vector space model**

PREPROCESSING:

1. Each line of the file is pre processed individually
2. Word tokenization is done using nltk library using the regextokenization which handles the formation of the tokens and also the removal of the punctuations
3. Stopwords have been removed from the tokens formed.
4. Lemmatization have been performed over the tokens using the nltk library
5. If the line contains any numbers, it is converted into words using inject library and is stored in the vocab along with the number itself

METHODOLOGY:

1. Traverse through all the documents, preprocess the data and maintain a vocab dictionary for each documents having word as the key and corresponding term frequency of the word in the document.
2. Create a list of dictionaries, having document name as the key for each dictionary and vocab dictionary of the corresponding calculated in the previous step as the value.
3. Maintain a list of unique vocabulary terms across all the documents
4. The query is then matched against the titles of the documents present in the index.html document. If there is 30% match of the query words with the title of the documents, the documents name is appended in the final document list
5. The query vector is calculated with its component value as the idf score corresponding to the terms present in the query, otherwise 0 for the terms not present in the query.
6. For each documents, document vector is created with the term frequency as the as the component against the unique vocabulary list of vocabs
7. For each documents, calculate the discreate cosine score by eucledian normalized form of the two vectors and append the score in the list
8. Sort the list based cosine scores.
9. For displaying the top k relevant documents, firstly the documents present in the match with the title displayed and remaining documents among the k documents is displayed from the sorted list of documents having cosine score
10. OBSERVATION

|  |  |  |  |
| --- | --- | --- | --- |
| SN.NO | QEURY | NUMBER OF WORDS | RETRIEVED DOCUMENTS |
| 1 | Disco can be fun | 5 | disco.be.fun, discocanbefun.txt, index.html, alissadl.txt, beyond.hum |
| 2 | Going 100 west by 53 north | 5 | 100west.txt,  arctic.txt, yukon.txt, charlie.txt, social.vikings |

**INFERENCE:**

There has been a difference in the number of documents fetched for the two methods. The difference is because of the fact in case of normal tf-idf ranking more weightage is on the tf score, where as in vector space mode both the scores are normalized by eucledian normalization. So if a term appears in the document is high but the idf value is less then it will appear in the top rank in case of the td-idf score but not in case of the vector space model.

As a result the vector space model is better than normal tf-idf score as it normalises both the scores giving equal weightage on both the scores and maintaining a uniformity across all the terms of the query