**INFORMATION RETRIEVAL**

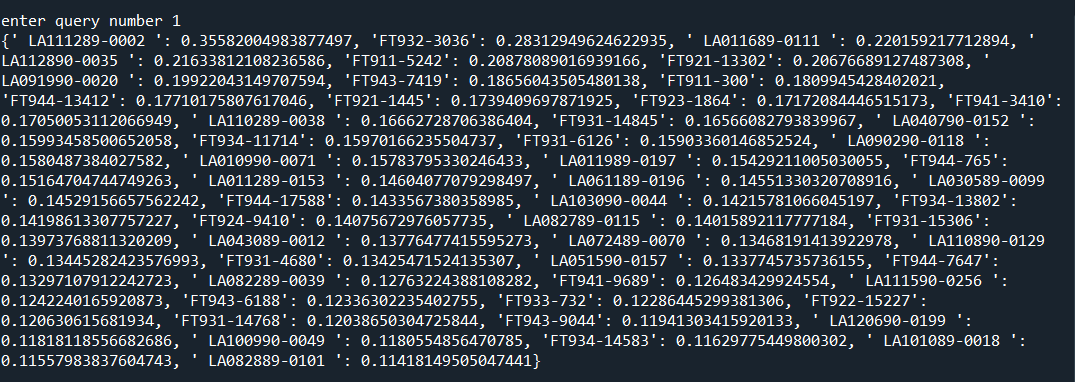
# REPORT:

## Baseline Document Retrieval Model:

Concept:

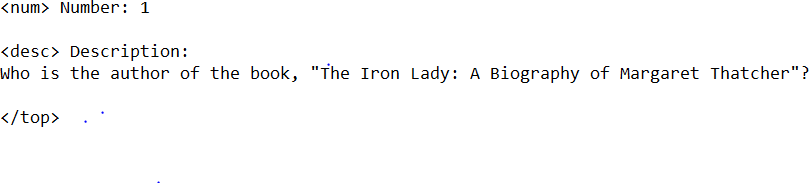
Here we utilise the concept of vector space model for document retrieval. Documents and queries are represented in a high-dimensional space, in which each dimension of the space corresponds to a word in the document collection. The most relevant documents for a query are expected to be those represented by the vectors closest to the query, that is, documents that use similar words to the query. To do retrieval in the vector space model, documents are ranked according to similarity with the query as measured by the cosine measure.

Output Screenshots and Explanation:

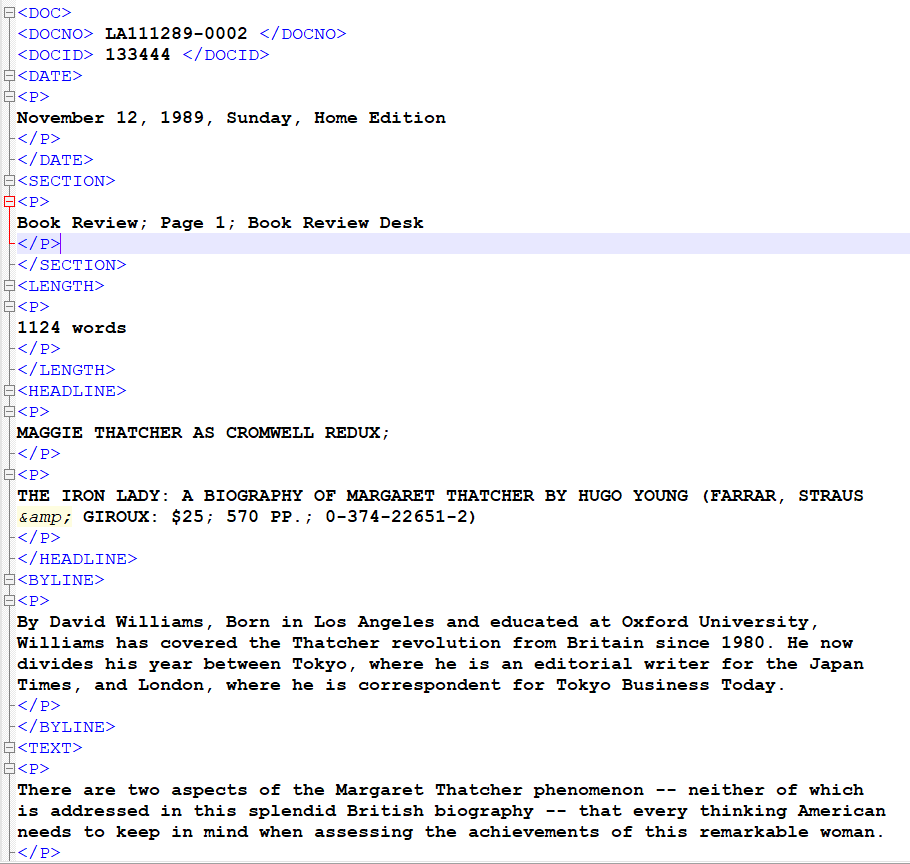


Analysis:

We have entered query number 1

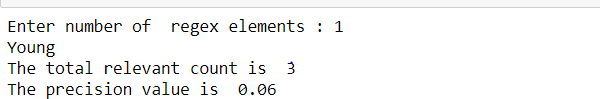


Here the first retrieved document is document ID: LA111289-0002 with score of 0.3558. Also, the corresponding query is: Who is the author of the book, "The Iron Lady: A Biography of Margaret Thatcher"? The below screenshot depicts the correct document retrieved with relevant to the given query number 1



* 1. Precision calculation

Calculation of evaluation metric precision at r= 50.Here we consider query 1 and for the query one the regex pattern to check is Young



Steps of procedure:

1. Import necessary packages
2. Extracting the text and document numbers from the corpus. Here we use BeautifulSoup library to parse the files.
3. Pre-processing the text using nltk libraries to tokenize, lower case and remove punctuations.
4. Compute and store the inverse document frequency(idf) of the terms.
5. Calculate the term frequency of the terms per document. We have implemented a function named **calculate\_tf\_per\_document(word\_list)** to implement this.
6. Compute and store the tf-idf weights for the terms. For this we have implemented a function namely **calculate\_tf\_idf\_per\_document(tf\_list)**
7. Extract the query file consisting of questions using BeautifulSoup library.
8. Pre-process the query file.
9. Compute the tf-idf for query file. We have implemented a function

**calculate\_tf\_idf\_per\_query(tf\_list)** to calculate this.

1. Vectorizing every document and for this we have implemented a function called **vectorize(every\_document)**
2. Compute cosine similarity scores, for this a function named **calculate\_cosine\_similarity(document\_key, query\_number)** is implemented.
3. Input the query number from the user
4. Displaying top 50 sorted cosine similarity scores along with document IDs
5. Compute and evaluate the performance of the document using the evaluation metric precision at r with r = 50
6. Input relevant regex pattern and also initialize count variable to zero. 16.Calculate precision value and display the corresponding result.

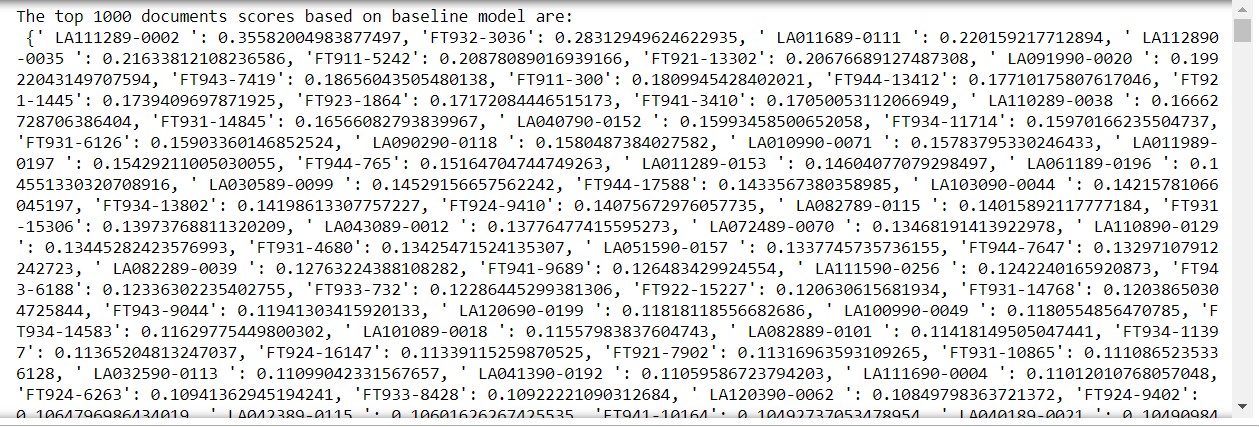
## Advanced Document Retriever with Re-Ranking

Concept:

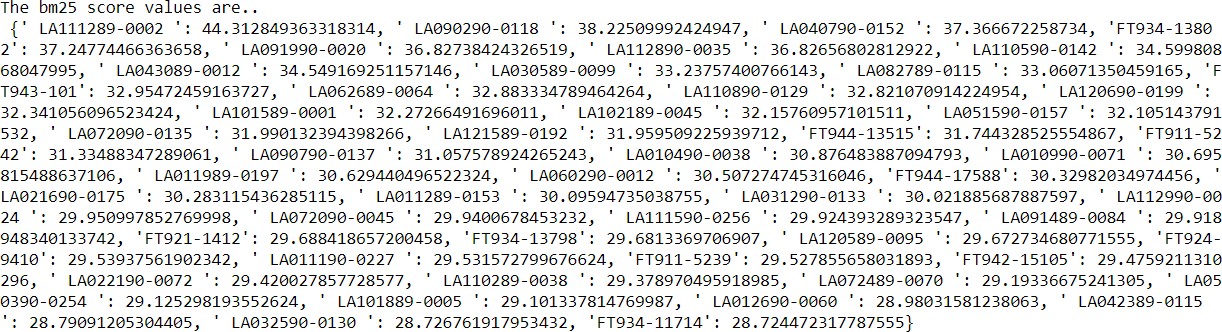
The BM25(Best Match) is a ranking function used to estimate the relevance of documents to a given search query.BM25 is bag of words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of their proximity within the document.

Output Screenshots and Explanation:

1. The top 1000 documents retrieved based on baseline model

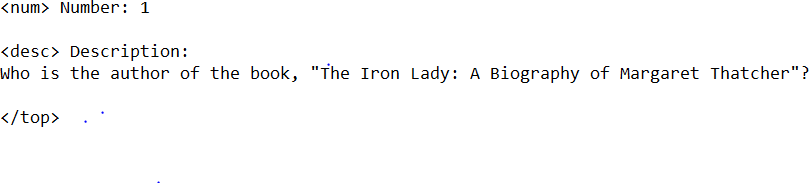


1. The top 50 documents retrieved based on BM25 scores



Analysis:

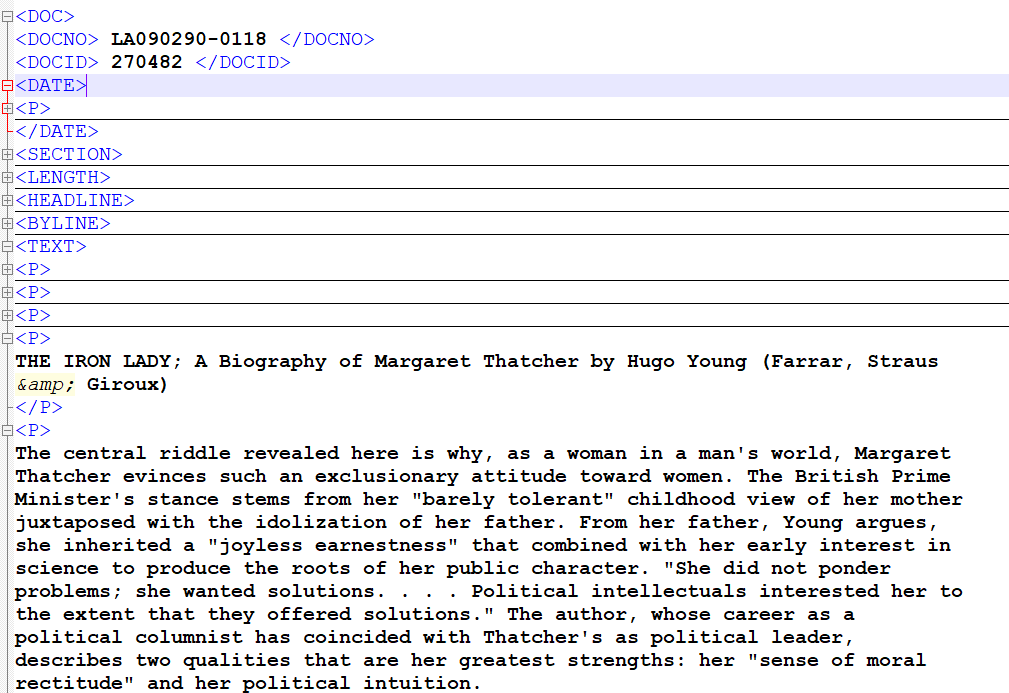
We have entered query number 1



Here the first retrieved document is document ID: LA111289-0002 with score of 0.3558. Also, the corresponding query is: Who is the author of the book, "The Iron Lady: A Biography of Margaret Thatcher"?

The below screenshot depicts the correct document retrieved with relevant to the given query number 1

Also, as per retrieved scores, the second most relevant document is ID number: LA090290-0118 and for this ID also the document text matches with the given query.



Steps of procedure:

1. Utilise the code from section 1 and then retrieve and display top 1000 sorted cosine similarity scores along with document IDs
2. Now we re-rank top 1000 documents with an advance model namely BM25 (Best Match technique) and retrieve top 50 documents.
3. Import genism package in order to use BM25
4. Pass the corpus to **BM25()** method and input the query to the

**get\_scores()** method

1. Sort the scores and display top 50 documents.

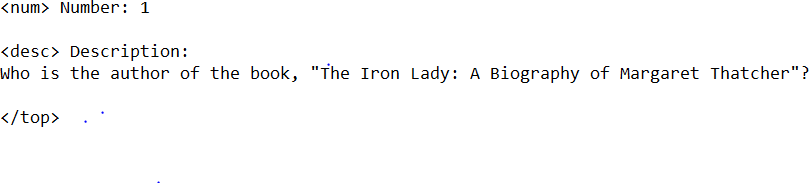
## Sentence Ranker

Output Screenshots and Explanation:

* 1. and (b) part:

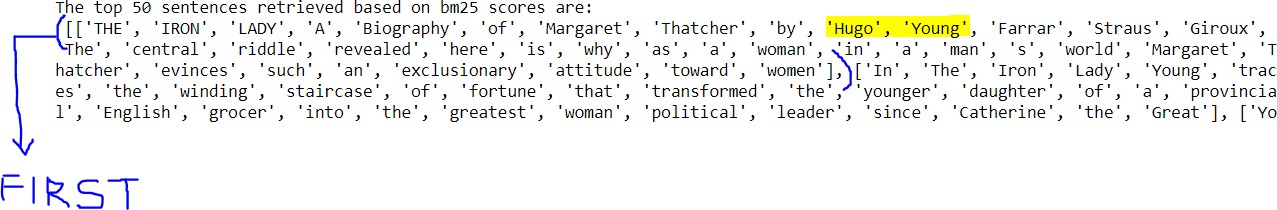
Analysis:

We have entered query number 1



Here, the first retrieved sentence:

THE IRON LADY; A Biography of Margaret Thatcher by Hugo Young (Farrar, Straus &amp; Giroux)



**matches** with the given query Who is the author of the book “The Iron Lady: A Biography of Margaret Thatcher”