Ranker Readme file (CS410 – Group MLTK)

This module is primarily associated with bringing out top 'k' tweets from the document collection.

As an example, to understand and illustrate the significance of this module, consider the query "Call of Duty". The harvester picks up the following tweet:

"While the Home Secretary @SuellaBraverman uses the report to call the nonviolent activists "extremists" and accuses the police of ‘institutional reluctance’. Telling them it is their ‘duty’ to take harsher action. https://t.co/Tgt9som2Sm"

But, this tweet shouldn't actually contribute to the 'sentiment' of call of duty in twitter. The ranker looks at the entire collection and understands the general context to de-prioritize this document. And that is evident in the output that the ranker produces. Thus, this module helps in filtering out tweets that may not actually be related to context which we are trying to analyze.

Multiple methods were used to score the relevance of each document and the best one was used. It is a normalized sum of products of the Term frequency and inverse document frequency of all words in a tweet.

The baseline weighs each word as the following:

$$TF\ (w) = \log(c(w,d) + 1)$$

$$IDF\ (w) = \log\left(\frac{M+1}{k}\right)$$

where *c(w,d)* represents the number of occurences of w in the document

and *k* represents the number of documents that contain the word 'w'

while, *M* represents the total number of documents in the collection

The final score of any document is calculated as follows:

$$Score\ (d) = \sum_{all\ words\ w\ in\ d} TF\ (w) * IDF\ (w)$$

$$Normalised\ Score\ (d) = \frac{Score(d)}{len(d)}$$

len(d) represents the number of words in the document. This normalization is performed to eliminate any length bias the collection may have.

The output is written into ../results/RankerOutput.txt, which will be fed to the next module (the sentiment analyzer).