Natural Language Processing Unit 1 Evaluation

Date: 29th Aug 2015

Time: 9:00 am to 13:30 pm IST

Problem # 1: Given a query string q and a corpus of documents, retrieve the top k documents that are the closest match

You are provided with a dataset that has a list of cricket commentary units. A single unit of cricket commentary is the commentary for 1 ball and this constitutes 1 document. For example, if your dataset contains the commentary for 300 balls it has 300 documents. An example of a single document (that is commentary for 1 ball) is as below:

"Meth to Jahurul Islam,no run,short of a good length and he leaves that one alone to the keeper"

Given a query string q you are required to return the top 10 matching documents as a Python list of document strings in the ascending order of rank. That is, you are required to return the list of the form: [d1, d2, d3, d4, d5], where d1 is the document that has the rank 1 and so on.

1. Your program should accept the query at run time using raw_input(). The examiner will provide the query string at runtime.
2. You are required to do the necessary preprocessing of both the query string as well as the documents in order to get them to a normal form. This preprocessing may involve (as you deem necessary):
    a. Tokenization (Sentence and Word levels)
    b. Case conversions
    c. Stop-word removal
    d. Lemmatization or stemming

    You may use NLTK APIs **ONLY** for this step
3. You are required to implement support for some basic semantics and at the minimum handle the following queries that are equivalent:
    a. If the query contains the term **boundary** it should match both FOUR and SIX.
    b. The term couple may correspond to 2 runs and should be treated equivalent
    c. The numeric value of FOUR (or any such word that corresponds to a number) should match the numeric value 4 also and likewise for other words that represent numbers

4. Pre-compute the term frequency scores for the documents. This also serves as the index of different tokens in the document.
5. You are required to handle exclamation marks and question marks that may be found in the query
6. You are required to compute TFIDF scoring (as per the equation as in Fig 1 below) for the normalized query string against each normalized document in the corpus
7. Return the top 10 documents ordered as per the rank, where the first document in the list is the document that has the top rank
8. You should handle conditions like: If the query contains any word that is not found in the document, you should leave that word out while computing the score. Implement this algorithm **without** add 1 smoothing. (If none of the documents in the whole corpus contains any word in the query string the score should be zero for that query for each document.)
9. We require that the implementation specified above returns documents that match the query where the query terms are considered as logical OR.
10. Modify your program to return ONLY those documents that match ALL the terms in the query (logical AND of query terms but NOT phrase matching). The query for such a match would be specified within double quote characters. That is, "q1 q2 …qk", where q1 to qk are the terms of the query placed within double quotes as shown.

    For example, if the query contains "Stuart Broad to Virat Kohli", you should return the ranked list of documents [d1, d2…, d10] only if every document di matches ALL the terms used in the query. Note that we don't need exact phrase matching. We only require that the given document contain ALL the terms in the query (regardless of the order) in order to be considered for ranking.

Deliverables:

1. Source code of your program
2. The test cases you used before submission. You are required to try multiple test cases on order to answer the point 3 below.
3. A brief write up on how effective tfidf was in the information retrieval pertaining to this problem and where did you find this algorithm not being optimal (This should be a FaceBook post within the deadline 30th Aug 2015 noon)
4. Any readme file or documentation that describes your work (optional)

Your submissions should be made to NLP course mail id ☺

Best of luck!

Figure 1:

$$\text{Score}(q, d) = \sum_{t \in q} \text{tf-idf}_{t,d}.$$