Massive Data Analysis Assignment 4

Questions: Finding Similar Articles

Given a set of BBCSports articles, Implement LSH using MapReduce to find out articles similarity.

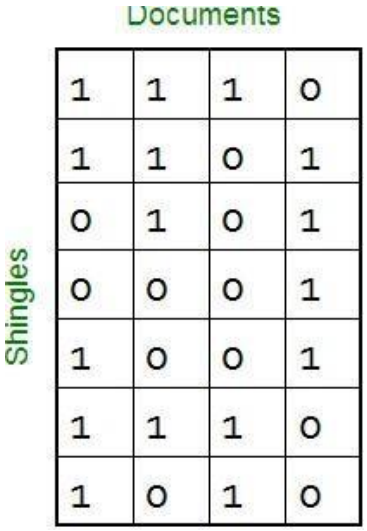
According to Ch.3, 3 essential steps are needed for similar docs. Thus, this assignment should contain these 3 parts.

1. **Shingling** : Convert documents to sets. Ch.3 part1 p.17~25

   The first part of this assignment is to implement the k-shingle, but this time we are going to use "words" instead of "characters".

   You should read in all the articles of the topic, and then translate them into shingles. After this part, your output may look like this example, which depends on how you implement your own Shingling.

   Note that you should use 3-shingles.



2. **Minhashing** : Covert large sets to short signatures, while preserving similarity.

   The goal in this part is that you should hash your shingles into smaller size to get "signatures".

   As we know, permuting rows even once is prohibitive because the cost is too high. However, there is a implementation trick.

As described in Ch.3 part1 p.43~p.44, instead of permuting rows, you can simply generate different hash functions to simulate the permutation step according to the algorithm.

Note that you need 100 different hash functions.

3. **Locality-sensitive hashing** : Focus on pairs of signatures likely to be from similar documents.

   Finally, we can apply LSH to the signatures. You should first partition your signature matrix M into b bands and r rows, where b = 50 and r = 2.

   Then, for each band, you should hash its portion of each column to a hash table with k buckets.

   The details are all in Ch.3 part1 p.53~60.

   Note that candidate pairs are those that hash to the same bucket for ≧ 1 band.

   After getting all the candidate pairs, you can calculate the Jaccard Simularity and then output the top 10.

Output format:

top 10 indices of candidate pairs and their similarities in decreasingly order.

(index1, index2) : similarity

Submit Requirements:

Upload **HW4_{studentID}.zip** which includes

a. **JAVA/python Code**
   Please make sure that your .java file has the same name as your class name, which must be LSH. If you implement the algorithm with Python, please name your .ipynb file as LSH, too
b. **Explain how you design every mapper and reducer.**
c. **Final output** (top 10 candidate pairs and their similarities)
   NOTE: Please sorted results decreasingly.