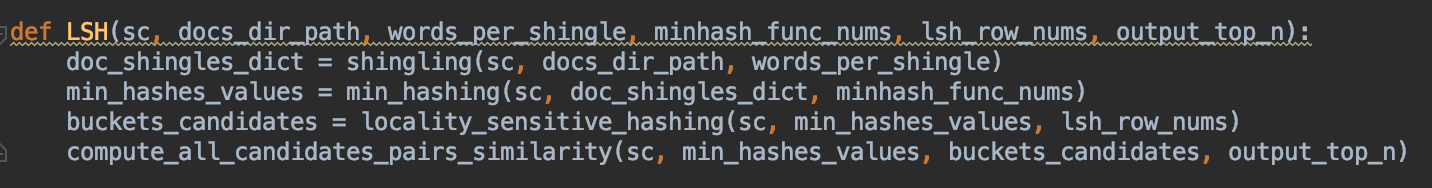
巨量資料 - 作業四

108062566 資工碩一 陳法佑

0. LSH由 Shingles, MinHashing, Locality\_sensitive\_hashing 組合而成



結果

(Doc\_idx1, Doc\_idx2): Jaccard Similarity

**(12, 20) : 1.0**

**(52, 84) : 1.0**

**(30, 35) : 0.75**

**(47, 49) : 0.73**

**(48, 49) : 0.55**

**(88, 49) : 0.44**

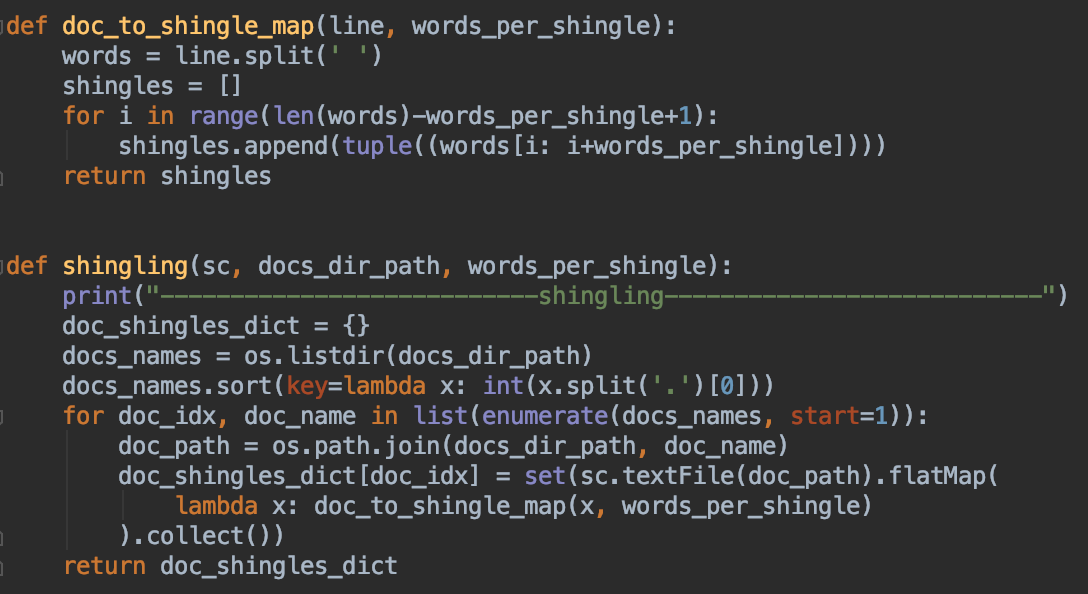
**(38, 23) : 0.43**

**(48, 47) : 0.42**

**(14, 40) : 0.36**

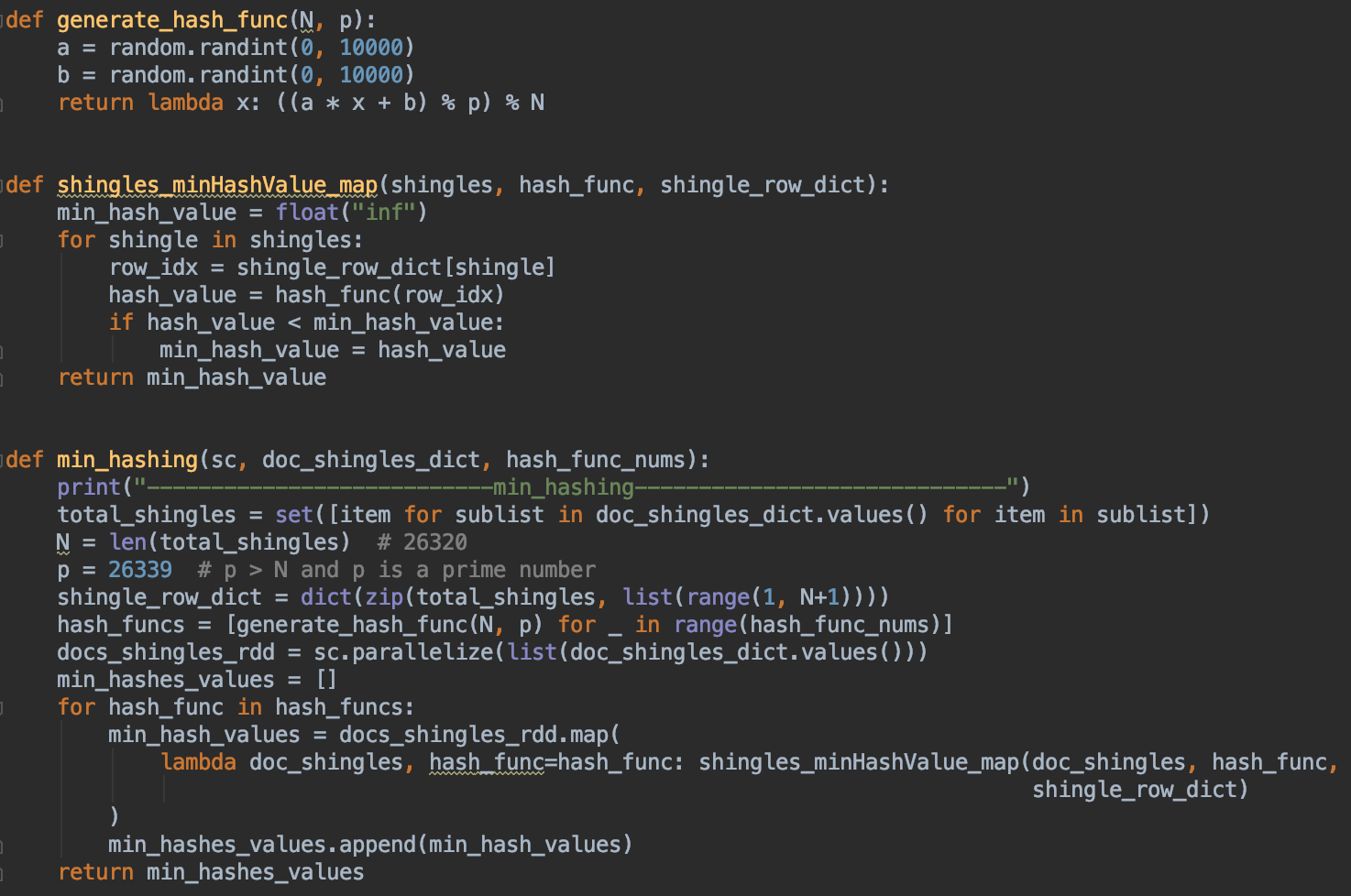
**(88, 47) : 0.3**

1. Shingling



將 documents 按照名稱排序後，用 map 對各個 document 搜集他們的 shingles，最後存入 dict。

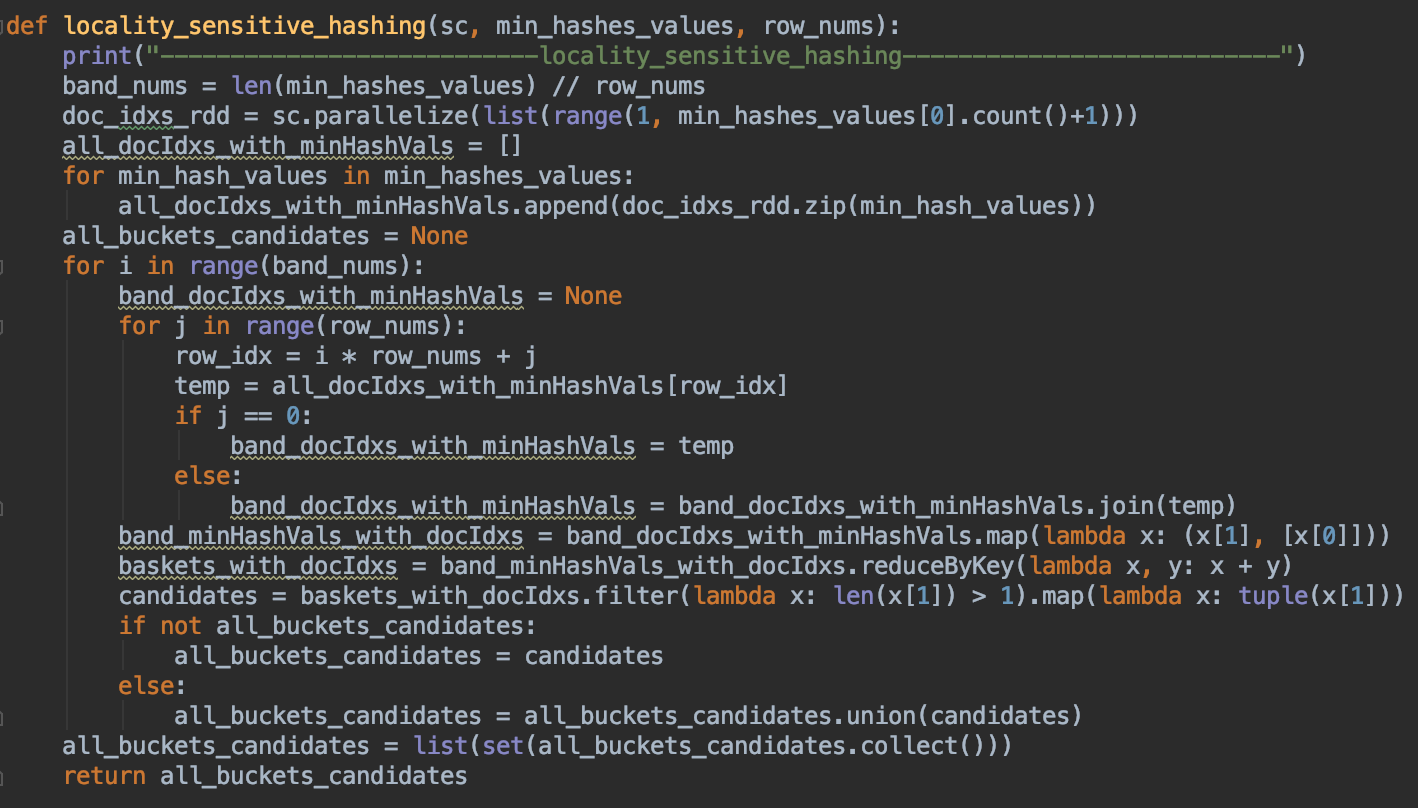
2. MinHashing



將所有 documents 的 shingles 搜集在一起並存成 set 的形式，然後將每個shingles 賦予一對一 1到N 的整數存成一個 「shingles\_row\_dict」，此處假設總共有 N 個 shingles。接著生成 100 個隨機的 Hashing Functions，再以 for loop 循環，每一次的 iteration 計算從「第一階段 Shingling」output 的 dict中所有 documents 的 minHash 值，計算時會用到 「hash\_function」

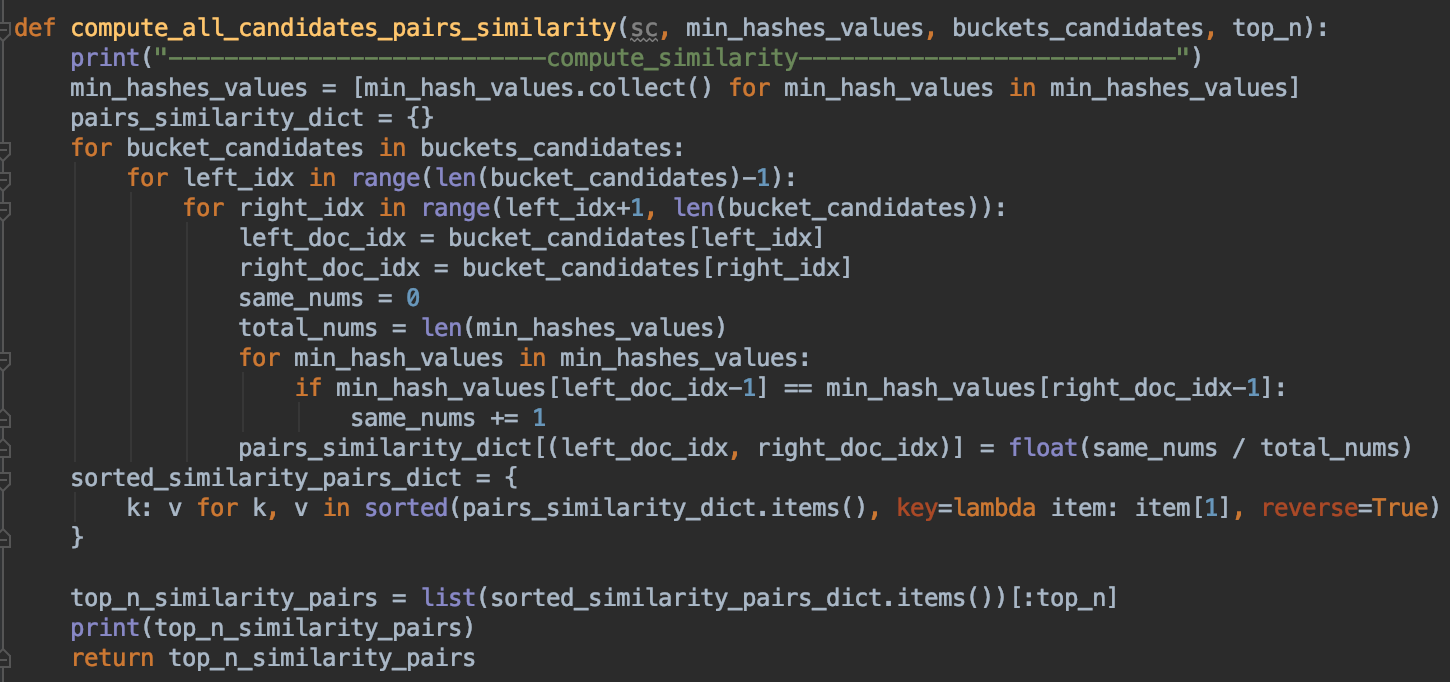
、「shingles\_row\_dict」及「一個 document 的所有 shingles」。而每一次的 iteration output 的維度是 (1, 101)，而總共 iterates 100次，所以最終 output 的維度是 (100, 101)。

3. Locality-Sensitive-Hashing



因為「第二階段 MinHashing」output 的「min\_hashes\_values」，他的 row 是「各個hash function」、col 是「各個 document」，但又因為「Locality-Sensitive-Hashing」需要用到同個 document 的兩兩 signatures，所以需要先做一個轉換。轉換方式是將「min\_hashes\_values」中每一個item都附上其 column index也就是 document index 的值，接著以 for loop iterates 「band\_nums」次，每一次將 「min\_hashes\_values」的2個 rows（題目要求） join 起來組合成「band」，此時資料形式為: [(doc\_idx, (row\_i\_min\_hash\_value, row\_j\_min\_hash\_value))…..]，接著將 key, value 互換順序，表示成[((row\_i\_min\_hash\_value, row\_j\_min\_hash\_value), doc\_idx)…..]，此時的 key 等同於一個「basket」，如果兩個 document 的 key 一樣，表示落入同一個 basket，因此實作中使用 「reduceByKey」實作此想法，最後再把 reduce 後有兩個以上落入到同一個 basket 的 document indexes 抓出來作為 candidates。

4. compute\_similarity



最後就是將落入同一個 baskets 的 candidates兩兩計算 Jaccard Similarity，再將相似度從高到低的印出。