Com S 535X Programming Assignment 2

Team: Pavithra Rajarathinam, Anubav

MinHash:

Terms Collection: To collect all the terms we iterate through each file in the folder and get the file content in a string. We remove the stop words and replace them with a space. Then the words separated by space character are considered as terms of the collection.

Data Structure:

To store all the terms in the document collection and give an id for each term	HashMap < Term, Term ID>
To store terms in each document a hashset is used which is used to form intersection	HashSet<TermID>>
To store the hashset corresponding to each document we have an array list of hashset	ArrayList <hashset>

Assigning Integer to Terms:

As each term of a document is processed, it is added to the hashmap with an id as per the order of insertion. Thus each term is given a unique id.

Permutation Used:

The permutations used is using permutation function.

Pseudo code:
exactJaccard

Get the file ID for both the filename
Load the corresponding the term ID hashset from the termID arraylist
Form the intersection of both the termID
Formulate Union Size = file1Set.Size + file2Set.size – intersectionSet.size
Exact Jac = Union Size / intersectionSet.size

minHashSig

Load the preinitialised random numbers a and b for each permutation function
Load the computed prime which is greater than the number of terms in document collection
Iterate through list of permutation function
    MinSig[permutation] = MaxInteger
    Iterate through the list of terms x in a document
    Compute hash = (a*x+b) % prime
    If MinSig[permutation] > hash
        MinSig[permutation] = hash

approximateJaccard

Find the fileid for the given files
Initialise similarityCount to 0
From the minhash matrix consider signature of two files and iterate through the values
Compare the minhash value for each permuation function
If they are equal then increment SimilarityCount
Else continue to the next value in signature
Approx Jac = SimilarityCount/ no of permutation

minHashMatrix

Calculate minHashSignature for each document
Form the matrix for entire document list.

MinHashAccuracy:

No Of Permutation	Error Factor	Mismatch Count
400	0.04	2742
400	0.07	5
400	0.09	0
600	0.04	418
600	0.07	0
600	0.09	0
800	0.04	72
800	0.07	0
800	0.09	0

Conclusion :
As the number of permutation increases the approximate Jaccard similarity is more accurate.

LSH:

To implement LSH, you need to create *b* (hash) tables, where *b* is the number bands. Though conceptually this is simple, this (may) present(s) a few implementation challenges.

- Java does not allow us to create an array of hash table as per the rule
  "An array creation expression creates an object that is a new array whose elements are of the type specified by the PrimitiveType or ClassOrInterfaceType. It is a compile-time error if the ClassOrInterfaceType does not denote a reifiable type
- So we created an array list of hash

• ***Hashing algorithm:***

Given a tuple i.e a band, we compute hash as
Initialise two random number a,b and a prime greater than 10 times the size of the permutation
Iterate through termID x in the tuple
HashInputStr = concatenate(HashInputStr, termID)
Hash = FNV(HashInputStr)

• ***Pseudocode for nearDuplciatesOf (filename)***

```
Initialise empty dupSet
Iterate through each hashtable h
        Iterate through entries x in the hash
        If set x contains the filename
                Add it to dupSet

Return dupList as an arrayList
```

NearDuplicates:

1.
For the doc space-0.txt bands 15 similarity of 0.9

Near Duplicates are - count 8 list [space-0.txt, space-0.txt.copy3, space-0.txt.copy2, space-0.txt.copy1, space-0.txt.copy7, space-0.txt.copy6, space-0.txt.copy5, space-0.txt.copy4]
After removing are 8 list [space-0.txt, space-0.txt.copy3, space-0.txt.copy2, space-0.txt.copy1, space-0.txt.copy7, space-0.txt.copy6, space-0.txt.copy5, space-0.txt.copy4]

FP = 0

2.
For the doc space-27.txt bands 15 similarity of 0.9
Near Duplicates are - count 8 list [space-27.txt.copy3, space-27.txt.copy4, space-27.txt.copy1, space-27.txt, space-27.txt.copy2, space-27.txt.copy6, space-27.txt.copy5, space-27.txt.copy7]
After removing are 8 list [space-27.txt.copy3, space-27.txt.copy4, space-27.txt.copy1, space-27.txt, space-27.txt.copy2, space-27.txt.copy6, space-27.txt.copy5, space-27.txt.copy7]

FP= 0

3.
For the doc space-27.txt bands 15 similarity of 0.9
Near Duplicates are - count 8 list [space-27.txt.copy3, space-27.txt.copy4, space-27.txt.copy1, space-27.txt, space-27.txt.copy2, space-27.txt.copy6, space-27.txt.copy5, space-27.txt.copy7]
After removing are 8 list [space-27.txt.copy3, space-27.txt.copy4, space-27.txt.copy1, space-27.txt, space-27.txt.copy2, space-27.txt.copy6, space-27.txt.copy5, space-27.txt.copy7]

FP=0

4.
For the doc space-45.txt bands 15 similarity of 0.9
Near Duplicates are - count 8 list [space-45.txt, space-45.txt.copy7, space-45.txt.copy5, space-45.txt.copy6, space-45.txt.copy3, space-45.txt.copy4, space-45.txt.copy1, space-45.txt.copy2]
After removing are 8 list [space-45.txt, space-45.txt.copy7, space-45.txt.copy5, space-45.txt.copy6, space-45.txt.copy3, space-45.txt.copy4, space-45.txt.copy1, space-45.txt.copy2]

FP=0

5. For the doc space-35.txt bands 25 similarity of 0.9
Near Duplicates are - count 8 list [space-35.txt.copy1, space-35.txt.copy3, space-35.txt.copy2, space-35.txt.copy5, space-35.txt.copy4, space-35.txt.copy7, space-35.txt, space-35.txt.copy6]

After removing are 8 list [space-35.txt.copy1, space-35.txt.copy3, space-35.txt.copy2, space-35.txt.copy5, space-35.txt.copy4, space-35.txt.copy7, space-35.txt, space-35.txt.copy6]
FP= 0

6.
For the doc space-35.txt bands 20 similarity of 0.9
Near Duplicates are - count 8 list [space-35.txt.copy1, space-35.txt.copy3, space-35.txt.copy2, space-35.txt.copy5, space-35.txt.copy4, space-35.txt.copy7, space-35.txt, space-35.txt.copy6]

After removing are 8 list [space-35.txt.copy1, space-35.txt.copy3, space-35.txt.copy2, space-35.txt.copy5, space-35.txt.copy4, space-35.txt.copy7, space-35.txt, space-35.txt.copy6]

FP=0

7.
For the doc space-45.txt bands 20 similarity of 0.9
Near Duplicates are - count 8 list [space-45.txt, space-45.txt.copy7, space-45.txt.copy5, space-45.txt.copy6, space-45.txt.copy3, space-45.txt.copy4, space-45.txt.copy1, space-45.txt.copy2]

After removing are 8 list [space-45.txt, space-45.txt.copy7, space-45.txt.copy5, space-45.txt.copy6, space-45.txt.copy3, space-45.txt.copy4, space-45.txt.copy1, space-45.txt.copy2]

FP=0